

# NUTS AND VOLTS

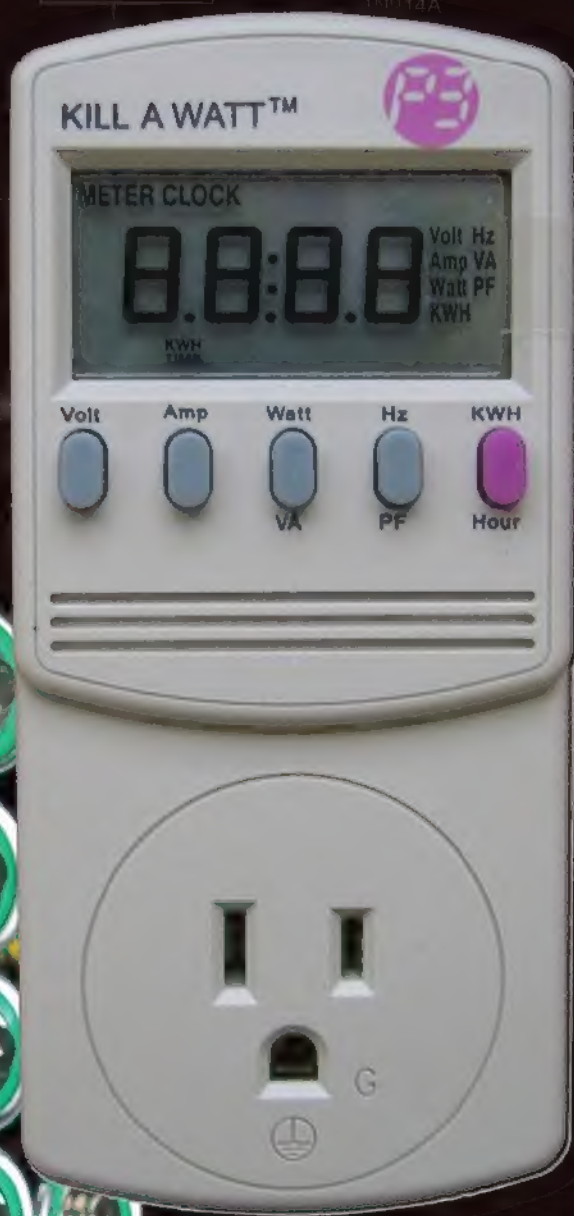
www.nutsvolts.com  
July 2015

EVERYTHING FOR ELECTRONICS

## HACKING The Kill-A-Watt

Overcome its low voltage limitations and create a useful test device for your bench.

- ◆ All About Ferrite
- ◆ FPGAs For The Hobbyist
- ◆ Reviving Vintage Stereo Equipment
- ◆ Power Sources - The Good, The Bad, And The Ugly





# STAY FOCUSED

Capture your world in high definition precision with our pocket-size MD10 action camera and the advanced resolution S60. Whether you are recording high adrenaline excitement with your friends or simply documenting workshop projects, these exceptional, lightweight cameras keep laser-sharp focus on all your fun!



MD10

#### FEATURES

- 3 Video Modes: 1080p30, 720p60, WVGA 120fps
- 8 Megapixel Still Images, 8fps Burst
- Intelligent Voice Control
- Optional IP-68 Housing, Waterproof to 20 Meters

#### TECHNICAL SPECIFICATIONS

- Recording Time (Max.)  
Video: About 210 Minutes  
Audio: About 240 Minutes
- Dimensions: 1.34 x 1.10 x 2.05 in.
- Weight: 1.76 oz.



S60

#### FEATURES

- 8 Video Modes: 720p120 ~ 1080p60
- 16 Megapixel Still Images, 8fps Burst
- 2-inch LCD Back for Viewing Footage
- IP-68 Housing, Waterproof to 10 Meters

#### TECHNICAL SPECIFICATIONS

- Recording Time (Max.)  
Video: About 180 Minutes  
Audio: About 210 Minutes
- Dimensions: 2.32 x 1.69 x 1.38 in.
- Weight: 1.92 oz.

**HITEC** | **AEE** | **Action Camera**  
Envision



# JUST HATCHED

But it's among the fastest breeds in existence

## FT900

32-bit MCU,  
speeds up to 310 DMIPS



## Baby Ostrich

Land bird,  
runs up to 70km/h



# July 2015



## 22 Modify a Kill-A-Watt EZ Power Meter for Low Voltage Operation

Discover a simple modification that safely powers the Kill-A-Watt meter for accurate operation in the low (10%-20%) voltage regime.

■ By Kevin J. O'Connor

## 26 Beyond the Arduino — Part 5

Please DO interrupt me! This time, take a look at how to handle interrupts in your Atmel AVR embedded projects, and how they can make your projects more efficient and easier to develop.

■ By Andrew Retailack

## 34 FPGAs for the Hobbyist

This tutorial will get you up and running with field programmable gate arrays — no prior experience required.

■ By Ryan Clarke

## 40 An Ultra Modern Shortwave Radio

A simple circuit, a USB TV tuner, your computer, and some powerful (free!) software combine to make an amazing software defined communications receiver for around \$25.

■ By George R. Steber

## 46 Reviving a Hi-Fi Classic: the Harman/Kardon A-401 Stereophonic Control Amplifier

Vintage stereo equipment just has a special sound and presence that are worth preserving. Follow along on a recent restoration of a 38 year old amp.

■ By J.W. Koebel

## 52 Power Sources: The Good, the Bad, and the Ugly

What you *don't* know about your project's power source may be the problem. Learn the most important features you should care about for four common power sources, how to measure them, and what to expect.

■ By Eric Bogatin



## Departments

05	DEVELOPING PERSPECTIVES <i>Change the World</i>	66	ELECTRO-NET
06	READER FEEDBACK	74	NV WEBSTORE
20	NEW PRODUCTS	77	CLASSIFIEDS
21	SHOWCASE	78	TECH FORUM
		81	AD INDEX

## Columns

### 08 Q&A

#### Reader Questions Answered Here

Topics this month include: a CR18650 battery vs. cell charging; help with vintage PA amps; and GFI breakers.

### 12 PICAXE Primer

#### Sharpening Your Tools of Creativity

PICAXE-PC Serial Communications Part 2.

We'll continue our exploration of hardware serial communication between a PICAXE and a PC.

### 18 Open Communication

#### The Latest in Networking and Wireless Technologies

Smartwatches Need Communications Too.

The new Apple Watch acts as a peripheral device to your smartphone by utilizing multiple wireless links between the two devices.

### 61 The Ham's Wireless Workbench

#### Practical Technology from the Ham World

All About Ferrite.

Rings, beads, cores ... find out exactly what ferrite is and what makes it ideal for a variety of uses in electronics.

### 68 The Design Cycle

#### Advanced Techniques for Design Engineers

Building a Multipurpose Communications Board.

Clear your bench and tin your soldering iron. In this installment, we're going to scratch build a multipurpose embedded communications board. While the soldering iron is cooling down, we'll fire up the Microchip XC32 C compiler under MPLABX and bring our new garage-brewed creation to life.





by Bryan Bergeron, Editor

# DEVELOPING PERSPECTIVES

## Change the World

**W**hile there's satisfaction from getting an LED to blink five times in a row in response to a button press, it isn't going to change the world — that is, unless you set your goals higher. It's easy to lose track of the fact that electronics have changed human existence in only a few decades. Computers, TV, satellites, space exploration, drones, cell phones, robots, aviation, and modern automobiles are but a few examples. Against that backdrop, there's a vast vacuum for experimentalists and engineers to fill. What I'm suggesting is that you take your knowledge of electronics — whatever your level of expertise — and focus it towards solving a meaningful problem.

I'm not suggesting you forgo experimenting with the basics such as triggering a microcontroller, but that you have a much larger purpose in mind — something to work toward other than simply acquiring knowledge for its own sake. For better or worse, there seems to be no end of problems to be solved.

Take the recent drought in California that negatively affects everyone in Silicon Valley. How can you leverage your knowledge of electronics to solve that dilemma? I don't have the answer, of course, but I'd start with looking for energy efficient means of desalination, perhaps some means of remotely monitoring the hydration of crops, and perhaps image processing techniques that can be applied to readily available satellite telemetry to track water consumption.

I'm sure you can think of a few dozen other areas to explore. The point is, there is no end of problems that you can address with your skills as an experimentalist. History suggests that your odds of changing the world are better if you have a clear vision of what you're going to accomplish, versus simply stumbling upon it. Of course, there are examples of accidental discoveries, but even then you have to be aware of the problems that need solving.

You might be reading this thinking what can you do at night in your basement, armed with a soldering iron and a few hundred parts? Well, others have done a great deal with much less computing power than is available on a common microcontroller. But it's a fair comment. Fortunately, thanks to the Internet and other communications means available, there's no need to go it alone.

Start something or find a group with a worthwhile

cause and join it. If money is a hurdle, then consider crowdfunding. Look for internships or summer jobs with companies addressing problems you want to solve or that at least get you part of the way there.

Finally, if you're young, seek out a mentor. I've been fortunate to find several over the years — each with a different perspective and skill set. Conversely, if you're older and experienced, become a mentor. Leverage your experience with a younger, less experienced and possibly more energetic student. Either way, it tends to be a win-win situation. Go ahead, change the world. I dare you.

**NV**

**WORLD'S MOST VERSATILE CIRCUIT BOARD HOLDERS**

Model 324 Our Circuit Board Holders add versatility & precision to your DIY electronics project. Solder, assemble & organize with ease.

Model 201

VISIT US ON  

**MONTHLY CONTEST**  
Visit us on Facebook® to post a photo of your creative PanaVise project for a chance to win a PanaVise prize package.

**PANAVISE**  
Innovative Holding Solutions

7540 Colbert Drive • Reno • Nevada 89511 | (800) 759-7535 | www.PanaVise.com



# READER FEEDBACK

## Happy About Ham

**A**s an amateur radio "nut"/licensee, I am extremely happy to see the recent wireless/ham radio content – please keep the articles coming!

After having had a long break from electronics over the past two decades due to life changes (and IT as a career), I am back into my ham

projects with a vengeance.

As well, the "back to basics" articles on the ATmega328 chip has been very helpful. One stumbling block I have had was over the plethora of development boards/shields. It is very timely to see this information presented at the "bare bones" level.

Due to many of my ham radio project interests involving Arduino and other micros, both of these subjects are very valuable to me.

Along the lines of keeping things simple, your excellent articles/information fit in very well with my interests in QRP DIY radio building and experimentation (less is more). Many thanks for the excellent publication!! 73s,

Ronan McAllister KB6NHQ

## Restoring a Quest

**R**egarding the recent "Fix Up That Old Radio" article in the May 2015 issue ... guys, you have no idea how long I've needed something like this. Back in 1966 when I was nine, I started my bucket list – and that was to restore my grandma's 1939 three-band Zenith console. I went as far as asking permission to inherit it when I grew up. I can still remember the startled look on her face as she smiled and said yes. I have since over the years, collected eight additional sets of varying makes and models awaiting their resurrections. Please keep up the good work and consider making "valve studies" a permanent feature.

I entered electronic studies and work some 25 years back (way after tubes were out). Solid-state was waning and digital was in its infancy, but I kept hoping one day to back track and complete my quest. As my kids used to say "are we there yet?"

G. Raines

## A Different Version

**R**ecently, a thunderstorm knocked out my barograph (appearing in the April 2015 issue), and in the course of fixing it I found that the updated Arduino IDE (Version 1.6.1) coughed out a spurious error message when compiling the graphical LCD library. After some puzzling, I downloaded the 1.0.6 version in which I developed the original code, and the error disappeared. Readers attempting this project – or any Arduino project

## ALL ELECTRONICS

www.all-electronics.com

Order Toll Free 1-800-826-5432

### 12MM PIEZO ELEMENT

with two 1" long leads.

CAT# PE-60

10 for 90¢ each

\$1.00 each

### PRO X FADE CROSS FADER

Eclectic Breaks (EB) 50K Ohm Linear x 2. A high quality retro-fit alternative to existing cross faders fitted in retail mixers. Users can physically change the feel of the fader to suit their own needs, by using adjustable cut-in rotaries and tension torque control. Easy access for lubricating and cleaning. CAT# SLP-50KL

\$9.50 each

10 for \$9.00 each

### 2-CONDUCTOR WATER-PROOF CONNECTOR

Interlocking 2-conductor bullet connector. 12" red and black leads, AWG 10 stranded copper. CAT# CON-321

\$3.00 each

10 for \$2.75 each  
100 for \$2.25 each

### DE-SOLDERING WICK

No-clean flux coated braid for easy removal of parts from P.C. boards. Top quality solder wick. Does not leave behind ionic flux residues. 5 feet per spool. Rohns compliant. Meets MIL-F-142560D.

Techspray 1822-5F #3 Green  
0.075" (1.9mm) wide.

CAT# SWK-3

\$3.50 each

25 for \$3.00 each

Techspray 1825-5F #6 Red.  
0.193" (4.9mm) wide.

CAT# SWK-6

\$4.00 each

25 for \$3.50 each

### "SOFT TOUCH" TACTILE PUSHBUTTON, SPST

Alps# SKEYAB028A. S.P.S.T., normally-open. "Soft-feeling" rubberized pushbutton for pc or breadboard mounting. 7.8 x 7.8 x 5mm. 1mm travel. Orange button on black base. CAT# MPB-143

4 for \$1.00

100 for 20¢ each

### 5VDC 2A POWER SUPPLY, USA/EURO/UK PLUGS

Input: 100-240 Vac, 50/60 Hz 0.4A.

Output: 5 Vdc, 2A.

6' cable with ferrite bead and 2.5mm coax plug, center positive. Includes three snap-on / snap-off AC plugs for use in USA, Europe and the UK. CE, cULus, TUV. CAT# PS-528

\$7.50 ea.

100 for \$6.75 ea.

### 10 MINI JUMPERS

10 test leads - 1" clips two each of 5 colors.

CAT# MTL-10

\$2.95 each

### 12VDC SPDT 40A AUTOMOTIVE RELAY

12 Vdc, 88 ohm coil. Contacts rated 40A. Plastic enclosed relay, 1.1" x 1.2" x 1" high. Plastic flange for bulkhead mounting. Mounts in standard automotive relay socket. CAT# RLY-351

\$2.55 each

10 for \$2.25 each



# NUTS AND VOLTS

EVERYTHING FOR ELECTRONICS

Published Monthly By  
**T & L Publications, Inc.**  
430 Princeland Ct.  
Corona, CA 92879-1300  
(951) 371-8497

FAX (951) 371-3052

Webstore orders only 1-800-783-4624

[www.nutsvolts.com](http://www.nutsvolts.com)

## Subscription Orders

Toll Free 1-877-525-2539

Outside US 1-818-487-4545

P.O. Box 15277

North Hollywood, CA 91615

## FOUNDER

Jack Lemieux

## PUBLISHER

Larry Lemieux

[publisher@nutsvolts.com](mailto:publisher@nutsvolts.com)

## ASSOCIATE PUBLISHER/ ADVERTISING SALES

Robin Lemieux

[robin@nutsvolts.com](mailto:robin@nutsvolts.com)

## EDITOR

Bryan Bergeron

[techedit-nutsvolts@yahoo.com](mailto:techedit-nutsvolts@yahoo.com)

## VP OF OPERATIONS

Vern Graner

[vern@nutsvolts.com](mailto:vern@nutsvolts.com)

## CONTRIBUTING EDITORS

Fred Eady

Lou Frenzel

Ward Silver

J.W. Koebel

Ryan Clarke

Kevin O'Connor

Tim Brown

Ron Hackett

Andrew Retallack

George Steber

Eric Bogatin

## CIRCULATION DEPARTMENT

[subscribe@nutsvolts.com](mailto:subscribe@nutsvolts.com)

## SHOW COORDINATOR

Audrey Lemieux

## WEB CONTENT

Michael Kaudze

[website@nutsvolts.com](mailto:website@nutsvolts.com)

## WEBSTORE MARKETING

Brian Kirkpatrick

[sales@nutsvolts.com](mailto:sales@nutsvolts.com)

## WEBSTORE MANAGER

Sean Lemieux

## ADMINISTRATIVE STAFF


Debbie Stauffacher

Re Gandara

Copyright © 2015 by T & L Publications, Inc.

All Rights Reserved

All advertising is subject to publisher's approval. We are not responsible for mistakes, misprints, or typographical errors. *Nuts & Volts Magazine* assumes no responsibility for the availability or condition of advertised items or for the honesty of the advertiser. The publisher makes no claims for the legality of any item advertised in *Nuts & Volts*. This is the sole responsibility of the advertiser. Advertisers and their agencies agree to indemnify and protect the publisher from any and all claims, action, or expense arising from advertising placed in *Nuts & Volts*. Please send all editorial correspondence, UPS, overnight mail, and artwork to: 430 Princeland Court, Corona, CA 92879.

Printed in the USA on SFI & FSC stock.  FSC

involving the graphical LCD libraries – may wish to roll back their Arduino IDE to this earlier version to avoid this compiler error.

Mark McGuire

## Back to the Future

The May 2015 issue of *Nuts & Volts* is by far my favorite issue ever. I love the stories about electronics of the past ... tubes, regen radio, bringing old electronics to life. Also, the basics – using a transistor as a switch.

I'm 63 and have been into electronics as an occupation and a hobby since the mid 1960s. I am certain that there are many readers – and potential readers – that have only a slight passing interest in microprocessors, Arduinos, Raspberry whatevers, and 3D printing. Older electronics is a market that is not being served by any other publication I know of. I know you cannot please everyone, but I think this is a huge area that needs attention. Many of us long for the days of *Popular Electronics* and *Radio Electronics* magazines.

I hope this trend in your magazine will continue.

Anthony Buzak

## Gravity Gradient Option

I love Paul Verhage's Near Space articles. To the April 2015 issue of *Nuts & Volts* article, "CubeSats – Part 3: Attitude and Velocity," I would like to add another method of satellite attitude control/stabilization: gravity-gradient stabilization. In gravity gradient stabilization, one of the satellite's dimensions is much longer than the others so that the satellite aligns its long axis vertical to the center of the body which it is orbiting (Earth, in the case of CubeSats).

The satellite is designed "bottom heavy" with the distance from its center of mass to a remote (not within the satellite) center of rotation such that the satellite acts as a pendulum whose period is equal to the orbital period of the satellite; the satellite will remain vertical to Earth's surface which is good for satellites with sensors that need to be pointed toward the Earth. This allows attitude stabilization without using fuel which reduces the weight of the satellite and allows more sensors to be installed.

Tim Brown PhD EE, PE  
N&V Q&A

## Feedback Motion Control

### The Old Way

- 1) Build robot
- 2) Guess PID coefficients
- 3) Test
  - 3a) Express disappointment
  - 3b) Search Internet, modify PID values
  - 3c) Read book, modify PID coefficients again
  - 3d) Decide performance is good enough
  - 3e) Realize it isn't
  - 3f) See if anyone just sells a giant servo
  - 3g) Express disappointment
  - 3h) Re-guess PID coefficients
  - 3i) Switch processor
  - 3j) Dust off old Differential Equations book
  - 3k) Remember why the book was so dusty
  - 3l) Calculate new, wildly different PID coefficients
  - 3m) Invent new, wildly different swear words
  - 3n) Research fuzzy logic
  - 3o) Now it is certainly not working in uncertain ways
  - 3p) Pull hair
  - 3q) Switch controller
  - 3r) Re-guess PID coefficients
  - 3s) Switch programming language
  - 3t) Start a new project that doesn't need feedback control
  - 3u) See parts in box. Feel guilty. Go back to old project
  - 3v) Start testing every possible combination of PID coefficients
  - 3w) Apply eye drops to red, bleary, sleep-deprived eyes
  - 3x) Wait, it's working!
  - 3y) Decide not to do any more projects that require control systems
  - 3z) Wonder why someone doesn't just make a thing that tunes itself

### The Kangaroo x2 Way

- 1) Build robot
- 2) Press Autotune
- 3) Get a snack

Kangaroo x2 adds self-tuning feedback to SyRen and Sabertooth motor drivers.



\$24.99

  
DimensionEngineering

[www.dimensionengineering.com/kangaroo](http://www.dimensionengineering.com/kangaroo)

July 2015 NUTS & VOLTS 7



In this column, Tim answers questions about all aspects of electronics, including computer hardware, software, circuits, electronic theory, troubleshooting, and anything else of interest to the hobbyist. Feel free to participate with your questions, comments, or suggestions. Send all questions and comments to: [Q&A@nutsvolts.com](mailto:Q&A@nutsvolts.com).

- **CR18650 Battery vs. Cell Charging**
- **Help with Vintage PA Amplifier**
- **GFI Breakers**

## CR18650 Battery versus Cell Charging

**Q** I have a low cost source of CR18650 cells. I would like to replace the NiCads in my power tools. Replacing the 3.6 volt versions in my old Black & Decker stuff is easy. Finding a charger is equally easy (less than a dollar for a USB one-cell charger). The problem is finding a solution for a 12 volt or 18 volt battery; specifically battery charging (e.g., it is easy to get rid of a few extra volts with some series diodes). I understand lithiums have "special" requirements such as over-temp monitoring, constant current, and constant voltage. However, I am willing to sacrifice charging time for simplicity/low cost. 73s.

— Bryan McPhee  
via email

**A** Lithium batteries are able to pack more electrical power into a smaller package which makes them desirable for electronics projects, but this also constitutes a danger of fire if the batteries are mishandled. Charging individual li-ion battery cells (three volts each) would be the easiest approach, but if you want to make a battery pack that can be charged as a whole you will have to use a charger designed for the major cordless power tool manufacturers (DeWalt, Makita, Craftsman, Bosch, Black & Decker, etc.). They make 18 volt li-ion battery packs for their power tools and also chargers for these battery packs (the Dewalt model DCB101 will charge 12 to 20 volt li-ion battery packs at a lot more than \$1 per cell though). The problem with using these chargers will be having the proper connections between the charger and your battery pack.

Li-ion batteries must be charged correctly, such as providing the correct current and voltage, not charging the battery to full (partial charge increases battery life), and stopping the charging process if the batteries reach a specific temperature. Manufactured chargers are design to take into account the needs of the li-ion battery. News headlines have shown laptops, cell phones, Boeing 787 Dreamliner battery bays, and even battery factories going up in smoke due to the energy density contained in the lithium batteries.

My electronics person half says build a charger yourself, but my engineer half says err on the side of safety and buy a commercial unit that has been certified. I don't want to see any of our readers on the 6 o'clock news.

## Help with Vintage PA Amplifier

**Q** I have a vacuum tube Executone amplifier I purchased awhile back. I would like to get it working, so I can play my electric guitar with it. My question is do you know of any good vacuum tube suppliers on the Internet and maybe a schematic for the amplifier. The model number and spec's are: PB-350, 50 watt, 115 volts AC, 60 cycles. It's missing four tubes.

— Brian J. Miller  
via email

**A** The PB-350 is an audio amplifier that was built by Executone, Inc., located in New York back in the 1950s. The PB-150 has five vacuum tubes: one 12AX7; two 12AU7s; and two 6550s (as per The Radio Museum's web page at [www.radiomuseum.org/r/executone\\_pb\\_350.html](http://www.radiomuseum.org/r/executone_pb_350.html)). [Note: One source says that there is a 5U4GB tube on the PA amp's chassis, but the 5U4GB seems to be used on the PB-235 and there may be a 12AU7 under the shield in the Executone amp.] As far as pricing, the 6550s will cost around \$40 each; the 12AX7 \$10; and the 12AU7s \$25 each, so just replacing the tubes will cost \$140. Check out [www.tubedepot.com](http://www.tubedepot.com) or [www.vacuumtubes.net](http://www.vacuumtubes.net) for vacuum tubes.

Add to this that all the electrolytic capacitors will need replacing along with resistors, potentiometers, non-electrolytic capacitors, tube sockets, wiring, and transformers. It could be bad news for your wallet. The PB-350 appears to actually be designed for public address systems (probably not so good for playing music), so it needs external speakers capable of handling the 50 watt output (another \$50 to \$100). There is a potential for a lot of expense to make this amplifier workable again.

By comparison, a 100 watt/dual channel Fender Champion 100 amp which has effects, tone/volume controls, two built-in speakers, and is designed/optimized



## QUESTIONS and ANSWERS

Post comments on this article at  
[www.nutsvolts.com/index.php?/magazine/article/july2015\\_QA](http://www.nutsvolts.com/index.php?/magazine/article/july2015_QA).

for use with a guitar is only around \$350 (close to the costs of tubes and two speakers).

The only inputs I see for the PB-350 are terminal strips labeled "Relay or Dummy Plug" and "Installation Connections," but descriptions say there is an RCA input and speaker tap outputs which may be 4/8/16 ohms or 70.7 volts

as used by PA systems. Look for a parts placement diagram and schematic inside the housing which was typically included in most vintage equipment. The schematic can confirm the type and number of tubes that are used and the input/output information. The parts placement diagram will show you the exact placement of the tubes.

I have included a sketch in **Figure 1** of the best tube placement I can determine. Some units have the tube identity printed on the chassis beside the tube location; if so, ignore my parts placement. Sorry, I could not find a schematic diagram.

You could always buy a new guitar amp and use the PB-350 as a conversation piece if all else fails, or unless you REALLY love working with vintage electronics.

Maybe some of our "vintage" readers can help with schematics, parts placement diagrams, etc.

### GFI Breakers

**Q** GFI breakers and outlet plugs function pretty simply. They compare current in with current out, and either open the circuit to protect the user or they continue to function. That's pretty much all I can find concerning GFI.

I have run into this situation now a couple of times, and cannot find an answer. Recently, I was changing plugs in a vintage 1978 home. It was wired in 12 gauge with a ground wire, and the box was populated with many circuits — all breakered at 15 amps. I changed a plug in a bathroom

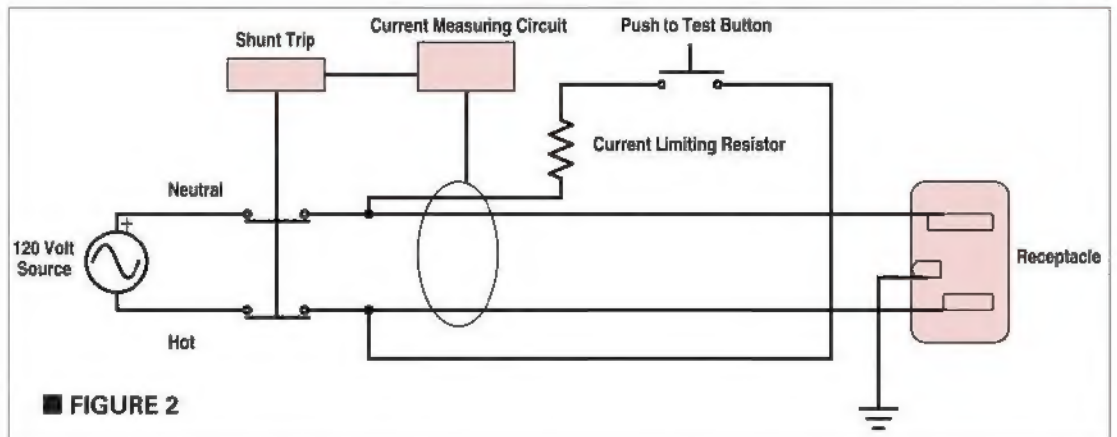
with a GFI typical box store item. The load side was in the proper connections which protected the second plug in this bathroom. Checking with a wiggy, all was well.

A day later, I get a call the plugs will not power a radio (a new off-the-shelf clock radio). I make another visit. Sure enough, the radio will work in a standard duplex receptacle but not the GFI. My assumption was I missed something. The wiggy worked. I tested with a carpet vac, a hair dryer, and a curling iron. They all worked, but not the radio. I have spent a while looking for an answer and nothing pops up. Pretty much every Google gives only how they work. My assumption is because the neutral has this GFI circuit blocking the direct flow back to the panel, some devices on two wire power supplies don't see the neutral and crap out. I have seen this with two wire power supplies on computers. The power supply will not run. I have seen this twice in the last year and nothing is written as to what the cause and effect is that I have found. Can you shed some light on this issue?

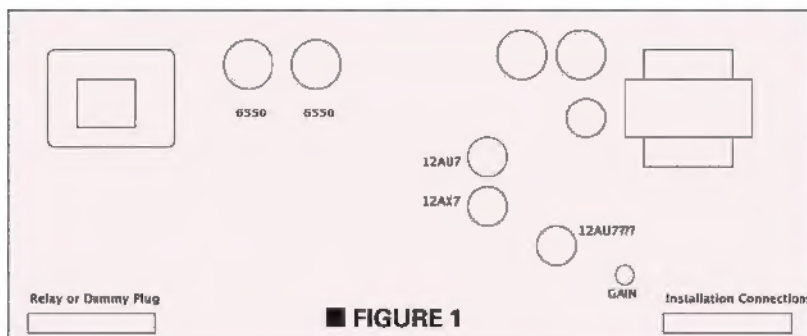
— Tim Edwards  
via email

**A** You are correct. The Ground Fault Interrupter (GFI) — also called a Ground Fault Circuit Interrupter; GFCI — is a very high speed switch (shuts down in 1/40th of a second) that is triggered by a current imbalance between the "hot" and "neutral" conductors of an electrical circuit. The imbalance is most often caused by current from the "hot" wire being diverted somewhere (such as a human body during electrocution), so a different amount of current flows in the "neutral" wire. GFCIs can be either mounted in place of a traditional receptacle or be placed inline with a power or an extension cord. GFCIs should be used anytime there is a possibility of a human making contact with the electrical device and a ground (usually water in some form).

The National Electric Code (NEC) requires GFCIs to be used to protect humans in bathrooms, kitchens, and swimming pools, so you are doing



■ FIGURE 2



■ FIGURE 1



the safe thing by installing a GFCI in a bathroom. The GFCI is effectively a straight through circuit with a switch that operates based on an unbalanced current in the hot and neutral lines (see **Figure 2**) which is the bulk of the circuitry and cost of the GFCI. So, for a normally operating electrical device, there should be no loss on voltage through the switch from the line to load terminals.

For readers who are not into the electrician side, a wiggly is an electrical test device with probes that indicates the presence of voltages between 120 and 600 volts AC

at 50 or 60 Hz, and 100 to 600 volts DC both visually and by vibration. The wiggly is not as accurate as a volt-ohm-milliammeter (VOM) or digital multimeter (DMM), but it is rugged and useful for detecting the presence of the correct voltage and checking fuses/breakers when you are not able to hold the probes and watch the meter (e.g., working inside a 480 volt three-phase electrical panel).

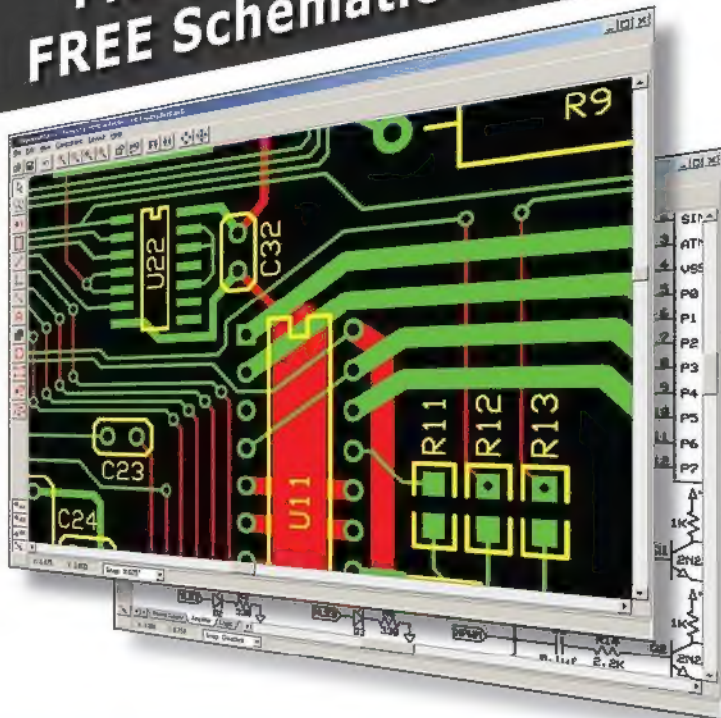
I looked up the current required for the devices you are plugging into the receptacle. They are: clock radio 0.067A; hair dryer 12.5A; curling iron 0.8A; carpet vacuum 4A. This represents quite a wide range of loads.

The first question I would have is what is the voltage (with a VOM or DMM since the wiggly only gives a rough value of voltage) at the line and load sides of the GFCI, and the voltage from the additional receptacle measured with and without a load on the additional receptacle. [I'm assuming you have checked the connections for security and have tried another set of GFCIs and receptacles.] Many times, an older home has wiring that has deteriorated, so the voltages are dropped due to additional circuit resistance. If there is resistance, it may not show up with no current flowing, but you will see a voltage drop once the load is drawing current (Ohm's Law).

Devices with motors (hair dryer and vacuum) can tolerate a voltage drop, and will have less power and possibly slightly lower speed which is most likely not detectable. Devices with resistance heaters (hair dryer and curling iron) will operate at lower voltages, but with a reduced heat which is probably not detectable either. Radios (particularly digital radios) convert the 120 volt line voltage to lower voltage DC levels using transformers, bridge rectifiers, and regulators, so there may be sufficient voltage loss from the already lower line voltage to the circuits (most likely IC chips) that the radio will not function properly.

Have you tried another radio? There may be a defect in the radio you are using, causing it to be intolerant to lower line voltages. **NV**

**\$51<sup>For 3</sup> PCBs**  
**FREE Layout Software!**  
**FREE Schematic Software!**

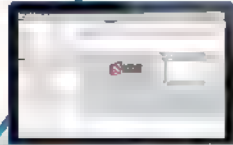


- 01 DOWNLOAD our free CAD software
- 02 DESIGN your two or four layer PC board
- 03 SEND us your design with just a click
- 04 RECEIVE top quality boards in just days

**expresspcb.com**



## The Easiest Way to Design Custom Front Panels & Enclosures



**You design it**  
to your specifications using  
our FREE CAD software,  
Front Panel Designer

**We machine it**  
and ship to you a  
professionally finished product  
no minimum quantity required

- Cost effective prototypes and production runs with no setup charges
- Powder-coated and anodized finishes in various colors
- Select from aluminum, acrylic or provide your own material
- Standard lead time in 5 days or express manufacturing in 3 or 1 days

**FRONT PANEL  
EXPRESS**

FrontPanelExpress.com

## Best Scopes, Best Prices



**Passport-Size PC Scopes \$129+**  
Great scopes for field use with laptops. Up to 200MHz bandwidth with 1GSa/s, high speed data streaming to 1MSa/s, built-in 1GSa/s AWG/wfm gen. PS2200A series



**30MHz Scope \$289**  
Remarkable 30MHz, 2-ch, 250MS/s sample rate scope. 8-in color TFT-LCD and AutoScale function. Includes FREE carry case and 3 year warranty! SD55032E



**100MHz Scope \$830**  
High-end 100MHz 4-ch 1GSa/s oscilloscope with 12Mpts memory, USB port, 7" display and Rigol's UltraVision technology. Includes FREE carry case! DS1104Z



**Versatile PC Scopes \$525+**  
2/4-ch 50-200MHz oscilloscopes with built-in function/arbitrary waveform gen and MSO options that combine fast sampling and huge buffer memory to 512Msa. PS3000D series



**70-300MHz Scopes \$839+**  
Fast, versatile 2-ch 2GSa/s scopes with 8-in WVGA LCD, integrated generator, 14Mpt memory, very low noise floor. FREE carry case available! DS2000A series

Professional Support  
24-Hour Customer Service



# JOIN TEAM SYNERGY MOON!

Synergy Space Explorers are the power that drives the Synergy Moon space program. We are Team Synergy Moon, building a mission to the moon that includes Micro Satellites, Lunar Rovers and a Lunar Lander. We're going into space this year, and we're going to the moon with our Google Lunar XPRIZE mission.



You will have the opportunity to participate in space research, exploration and development missions, which currently include our Google Lunar XPRIZE mission to the moon, the Artemis NanoSat Constellation project and our remote controlled Tesla Orbital Space Telescope.

**Get on board now and be part of this great adventure!**

**DIY SATELLITES AND SPACECRAFT SYSTEMS**  
info@synergymoon.com



# PICAXE-PC Serial Communication — Part 2

In the previous installment of the Primer, we built a simple stripboard circuit to interface the Prolific PL2303HX USB-to-serial cable with an 08M2 processor, and conducted four simple software experiments that explored the process of establishing a PICAXE-PC serial communication link via the PL2303HX cable.

In those experiments, we tested and compared the PICAXE *serin* and *hserin* commands, and demonstrated the advantages of using the *hserin* command for serial communication between a PICAXE processor and a PC. Specifically, we discovered (as long as we don't send more than two serial bytes within a short period of time), the *hserin* command is always able to capture the serial data in the background while the M2 program is busy doing other tasks.

This month, we're going to continue our exploration of hardware serial communication between a PICAXE project and a PC. We'll begin by using a 20M2 processor to essentially replicate our final experiment in the previous column. When we're sure our breadboard setup is functioning correctly, we'll move on to a 20X2 processor which will enable us to demonstrate the vastly increased power of its hardware serial commands, as compared to those of the M2-class processors. So, let's get started!

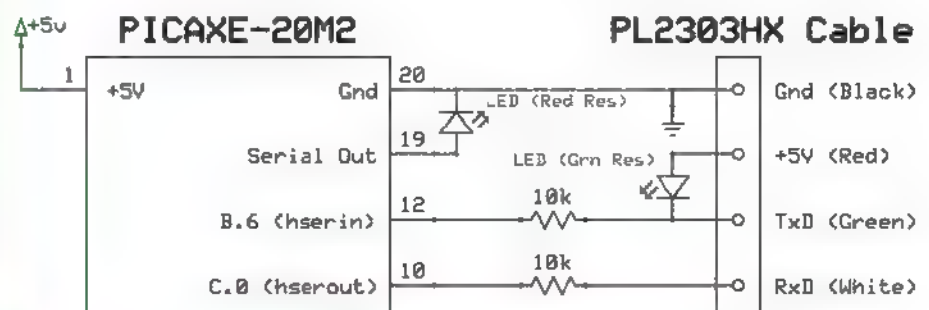
## Experiment 1: Using *Hserin* to Receive Serial Data in the Background

The schematic for our first experiment is presented in Figure 1. If you compare this schematic to the one we used last time (Primer May 2015, Figure 1), you can see that we're again using the same setup, but this time our processor is a PICAXE 20M2. On the 20M2, the *hserin* pin

is pin B.6, and the *hserout* pin is pin C.0. As a result, the TxD pin on the PL2303HX cable is connected to pin B.6 (*hserin*), and the RxD pin on the PL2303HX cable is connected to pin C.0 (*hserout*).

Figure 2 is a photo of my breadboard setup for this experiment. (The same breadboard setup will be used for all our experiments this month, but before we're done we'll replace the 20M2 processor with a 20X2.) As you can see in Figure 2, I'm using the same stripboard interface circuit that we constructed in the previous installment to interface the PL2303HX cable with an 08M2 processor. At that time, I mentioned that I actually had the 20M2 in mind when I developed the stripboard layout, which is apparent when you see how it is to interface the stripboard with the 20M2.

However, there are three minor details about the breadboard setup that I should mention: First, since the stripboard circuit does not include a connection to the ground rail, we need to install one on the breadboard so that the cable and the processor share a common ground; second, even though the gray-painted pin on the stripboard is inserted directly in line with pin B.7 on the 20M2, the pin on the stripboard is not electrically



■ FIGURE 1. Schematic for Experiments 1-4.



Post comments on this article and find any associated files and/or downloads at [www.nutsvolts.com/index.php?/magazine/article/July2015\\_PICAXEPrimer](http://www.nutsvolts.com/index.php?/magazine/article/July2015_PICAXEPrimer).

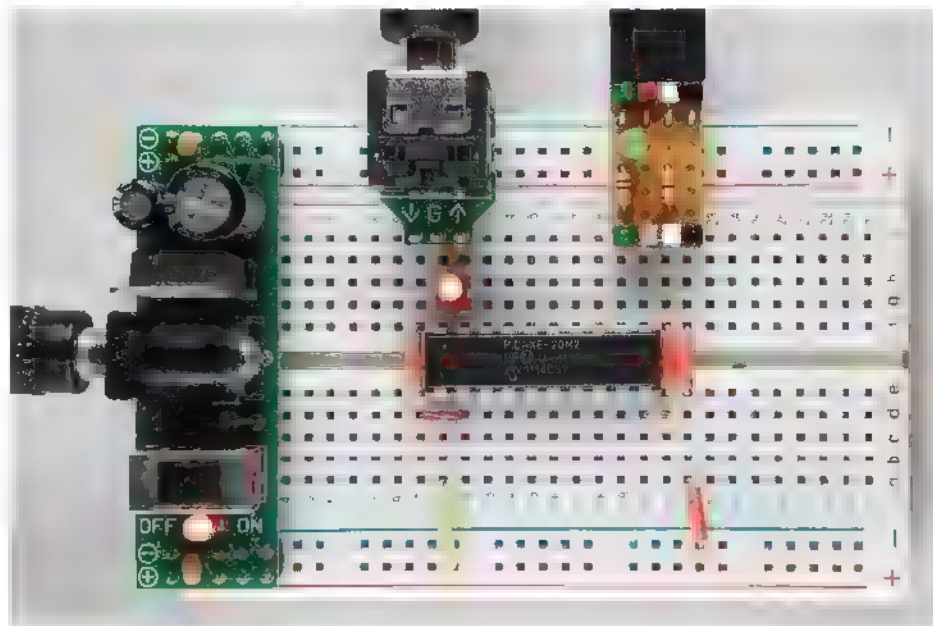
connected to anything, so pin B.7 can be used for any purpose you want without creating an electrical problem; third, don't forget to install the short bare jumper that connects the **RxD** pin on the PL2303HX to pin C.0 on the 20M2.

The software for our first experiment this month (*hserin20M2.bas*) is available for downloading at the article link. Now would be a good time to download it, along with the other programs we will be using this month. If you read through the program listing for the *hserin20M2.bas* program, you will see that — aside from the fact we're using a different processor — there are two minor differences from the *hserinFromPC.bas* program that we used in the previous column.

First, the pin assignments for the *RxPin* and the *TxPin* have changed. On the 20M2 processor, the *hserin* pin is pin B.6, and the *hserout* pin is pin C.0. We aren't using the *hserout* command in our first experiment, but we will in our second experiment. Using pin C.0 for the *serout* command will enable us to use the same hardware setup for the next experiment, as well.

The second difference has to do with the "undocumented A.0 pin" on the 20M2 processor. We've already discussed this issue (in the February 2012 Primer), so I'll just briefly re-state it at this point. The PICAXE compiler recognizes *A.0* as an undocumented name for the *Serial Out* pin on the 20M2 processor, and accepts a few undocumented "pseudo" commands, including *high A.0*, *low A.0*, *toggle A.0*, and *serout A.0*. Unfortunately, it's not possible to define a name for the undocumented *A.0* pin: including a *symbol LED = A.0* statement in a program produces a syntax error.

However, whenever we want to blink the LED on the 20M2 *Serial Out* pin, we can just use a combination of *high A.0* and *low A.0* commands (with suitable pauses), or even a *toggle A.0* command. (As we'll soon



■ **FIGURE 2.** Breadboard setup for Experiments 1-4.

see, the same approach also works with the 20X2 processor.)

With the above information in mind, let's replicate Experiment 4 from the last Primer. This time, however, we're using a 20M2 processor. You may want to refer back to the previous Primer installment for the details of setting up the experiment and your terminal program.

When you're ready to proceed, use **Figures 1 and 2** to assemble your breadboard circuit, and download the *hserin20M2.bas* program to the 20M2. The results you obtain by typing one or two characters into the terminal program should be the same as last time.

## Experiment 2: Adding *hserout*, and Speeding Things Up

In our previous hardware serial experiments with the 08M2, we weren't able to use the *hserout* command. As I explained then, on both the 08M2 and 14M2, the *hserout* command is only available on the processor's *Serial Out* pin (C.0 on the 08M2, and B.0 on the

14M2). As a result, if we were to use the *hserout* command, every time we downloaded a new program to the processor we would also be sending "garbage" characters out to the PC. In order to avoid that problem, we used the *serout* command with pin C.2 on the 08M2.

Fortunately, on both the 20M2 and 20X2 processors, the *hserout* pin does not conflict with the *Serial Out* pin. On both 20-pin processors, the *hserout* pin is pin C.0 and the *hserin* pin is pin B.6, which means that we're now free to experiment with the *hserout* command and speed things up a bit.

**Figure 3** presents the valid serial baud rates for hardware serial I/O, along with the baud rates that are supported by the CoolTerm program. (If you're using a different terminal program, you will need to check your documentation for the valid baud rates.)

For each baud rate, the entries in the middle column of **Figure 3** show the required value for the *baud\_setup* parameter of the *hsersetup* command. The "X" in each entry must be replaced by the appropriate value (in MHz) for the clock speed at which the PICAXE processor is running.



For example, if we want to run our processor at 8 MHz and set up our hardware serial connection at 9600 baud, the correct *baud\_setup* parameter would be *B9600\_8*.

Because we're going to enable the *hserout* command in this experiment, we need to modify the *hsersetup* command that we have been using so far. We're also going to speed things up quite a bit. If you refer back to **Figure 3**, you can see that 115200 baud is the fastest baud rate that's supported by both the *hsersetup* command and the *CoolTerm* program. Again, if you're using a different terminal program, refer to its documentation to determine the fastest baud rate that's supported by it and the *hsersetup* command. We'll be using the *hserial20M2fast.bas* program for this experiment, and our breadboard

setup remains the same as Experiment 1.

Before you actually run the experiment, let's take a little time to discuss the differences between the *hserin20M2.bas* program (Experiment 1) and the *hserial20M2fast.bas* program (Experiment 2). At this point, you may want to either print out both programs, or set up your PICAXE Editor screen so that you can view them both simultaneously.

In Experiment 1, there was no need for us to define an *RxPin* constant because on the 20M2 (and 20X2), that pin is fixed for hardware serial input (*RxPin* = B.6). In Experiment 2, we will be using the *hserout* command, so (as you might suspect) there's also no need to define a *TxPin* constant. (The *hserout* command can only be used on pin C.0 of the 20M2 or 20X2 processors.)

Because we're also using the *hserout* command this time, we need to change the *hsersetup mode* parameter. We're increasing the baud rate (dramatically) as well, so we need to change the *baud\_setup* parameter too. The complete syntax that we'll use is *hsersetup B115200\_4,%00000*. (The "0" in the bit 3 position of the mode parameter indicates that we want to enable the *hserout* command.)

At this point, I should also mention a convenient side benefit to the use of hardware serial communications. In Experiment 1, we had to run the 20M2 at 8 MHz in order to be able to use 9600 baud for our *serout* statements. (In fact, 9600 baud is the fastest possible baud rate for a 20M2 running at 8 MHz.) On the other hand, any M2-class

processor running at its default rate of 4 MHz is capable of hardware serial I/O at any of the baud rates specified in **Figure 3**.

The side benefit of this increased flexibility is the fact that we can run the processor at its default rate (4 MHz for M2-class processors), so we don't need to remember to adjust the timing of our *pause* and/or *wait* statements. For example, in Experiment 1 we were running the 20M2 at 8 MHz (twice its default rate), so we also had to remember to write *wait 2* for a one second delay. In Experiment 2, the 20M2 is running at its default rate (4 MHz), so we can simply write *wait 1* for a one second delay.

The only other difference between the two programs is the *hserout* command. Its complete syntax is *hserout break,({#}data,{#}data,...)*. The *break* parameter (0 or 1) is used to indicate whether or not to send a "break" (wake up) signal before the actual data is transmitted. We don't need to do that, so we'll use 0 as the value of the *break* parameter.

One final point before you test *hserial20M2fast.bas* on your breadboard setup: I tested it on three different 20M2 processors, and all three of them were able to send and receive the data at 115200 baud, which is pretty impressive. However, there's always a possibility that a slight frequency error in the internal oscillator of any M2-class processor could result in serial I/O errors. If that happens to you, try decreasing the baud rate a little to see if it corrects the problem. If it doesn't, you may need to troubleshoot your breadboard setup.

## Moving Up to the 20X2 Processor

As I mentioned in the last column, hardware serial communication with the 20X2 (as well as all the other X1- and X2-class processors) is much more powerful

<i>hsersetup</i> Baud Rate	<i>baud_setup</i> parameter	<i>CoolTerm</i> Baud Rate
300	**B300_X	300
600	B600_X	600
1200	B1200_X	1200
----	-----	1800
2400	B2400_X	2400
----	-----	3600
4800	B4800_X	4800
----	-----	7200
9600	B9600_X	9600
----	-----	14400
19200	B19200_X	19200
-----	-----	28800
31250	B31250_X	-----
38400	B38400_X	38400
57600	B57600_X	57600
115200	B115200_X	115200
-----	-----	230400

\*\* X = 4 for 4MHz, 8 for 8MHz, etc.

**FIGURE 3.** Baud rates for *hsersetup* and *CoolTerm*.



than what's available on the M2-class processors. To begin with, the 28X1, 40X1, and 20X2 processors are able to receive up to 128 serial bytes in the background at one time, and the 28X2 and 40X2 processors can receive as many as 1024 serial bytes in the background, while all M2-class processors are limited to just two bytes of background serial data.

In addition, on all X1 and X2 PICAXE processors, we can configure the reception of serial data so that there's no need to even include a single *hserin* statement in our program. In the following discussion, we'll focus on the 20X2 processor, but don't forget that all the information that we will cover also applies to all currently available X1 and X2 processors.

Previously (and in Experiment 1 this month), all our programs included the following program statement:

```
hsersetup B9600 8, %01000
```

Recall we discussed the basics of the *hsersetup* command, including the function of each of the five digits in its **mode** parameter (%01000 in the above statement). For easy reference in the following discussion, **Figure 4** summarizes the settings for each bit of the *mode* parameter.

At this point, let's focus on bit0 of the *mode* parameter. As I mentioned last time, all M2 processors **do not** contain a scratchpad memory area,

which is why we had to specify "0" for bit0 of the *mode* parameter. On the other hand, all X1 and X2 processors **do** contain a scratchpad, so we can set bit0 equal to 1. When we do that, we no longer need to use the *hserin* command in our program; any serial data that arrives is automatically stored sequentially in the scratchpad without any programming effort on our part. All we need to do is access and process the data that has been automatically received in the background.

There are two main approaches to storing or retrieving data in any area of a microcontroller's random access memory, including the 20X2 scratchpad: **direct** addressing and **indirect** addressing. Direct addressing is the more common approach, and it's what we use whenever we assign or retrieve a value for one of the general purpose variables. For example, when we write *b0 = 10*, we're directly storing the value 10 at the *b0* memory location.

Indirect addressing is much less straightforward, but it's also much more powerful (and faster) than direct addressing. We've discussed indirect addressing in three earlier Primer articles (October 2011, December 2011, and February 2014), and if we needed to store incoming serial data in the 20X2 scratchpad, we would also need to use indirect addressing again.

However, in the present context,

the *hsersetup* command enables us to automatically store the incoming serial data, so all we need to do is access and process it. Fortunately, those two tasks do not require the speed and power of indirect addressing, so this time we can simply use direct addressing.

There are three built-in variables that we need to use in order to access and process the serial data that has been automatically stored in the scratchpad:

- The *hserflag* variable is one of the bits in the *flags* system variable. It is automatically set to 0 when hardware serial data has been received in the background. When our program has finished processing the received serial data, we must reset *hserflag* to 0 in preparation for any subsequent serial input. It isn't mentioned in the documentation for the system variables (PICAXE manual, section 2, page 14), but *hserinflag* is also recognized as an alternate name for the same flag. Since the name *hserinflag* more clearly states its function, that's the name we will use in the remaining experiments this month.

- The *hserptr* variable is a system variable that points to the location in the scratchpad (0-127 for the 20X2) that will receive the next incoming hardware serial byte. When our program has finished processing the received serial data, we must

reset *hserptr* to 0 in preparation for any subsequent serial input. Again, it isn't mentioned in the documentation for the system variables, but *hserinptr* is also recognized as an alternate name for *hserptr*. Since the name *hserinptr* more clearly states its function, that's what we will use in the remaining experiments this month.

■ **FIGURE 4.** Settings for the *hsersetup* mode parameter.

Bit	Value	Function
bit0	0	Do not use scratchpad
	1	Receive background serial data to scratchpad
bit1	0	Do not invert serial output data
	1	Invert serial output data
bit2	0	Do not invert serial input data
	1	Invert serial input data
bit3	0	Do not disable hserout
	1	Disable hserout (hserout pin is normal I/O)
bit4	0	Do not disable hserin
	1	Disable hserin (hserin pin is normal I/O)



- The *ptr* variable is a special function variable that points to a location in the scratchpad (0-127 for the 20X2) that we want to access. The *ptr* variable can be used in conjunction with the *get* command which can be used to access (read) data from the scratchpad. In its simplest form (i.e., to read a single byte from the scratchpad), the syntax of the *get* command is as follows:

```
get location, variable
```

If you're interested in more information on using the *get* command to read multiple bytes with one statement and/or to read word variables, see the documentation for the *get* command in section 2 of the PICAXE manual. For our present purpose, however, let's look at a simple code snippet that reads one byte from the scratchpad (assuming that the *chr* variable has been previously defined in a *symbol* statement):

```
ptr = 0
get ptr, chr
```

The above code snippet simply reads the value that's currently stored in location 0 of the scratchpad, and then assigns that value to the *chr* variable for subsequent processing. Of course, it does get more complicated than that because we also want to be able to read a sequence of characters that have been automatically received into the scratchpad. To do so, we also need to correctly manipulate the values of *hserinflag*, *hserinptr*, and *ptr*, as we're about to see in our next experiment.

## Experiment 3: Moving Up to the 20X2 Processor

In this experiment, we'll explore the more advanced hardware serial I/O capabilities of the 20X2 processor, as compared to the 20M2

processor that we have already investigated. We're still going to use the CoolTerm program (or whichever terminal program you prefer), but in a slightly different way.

Previously, we configured our terminal program to operate in *Raw Mode*, which means that each character we type in the terminal is immediately transmitted to the PICAXE processor. This time, we're going to use *Line Mode*, which means that we can type a complete line of text, and nothing will be sent to the 20X2 until we press the *enter* (or *return*) key on our Mac or PC. In order to do so, we need to change a couple of settings in the terminal program.

First, change the terminal setting from *Raw Mode* to *Line Mode*, and make sure the *enter* (or *return*) key emulation is set to *CR+LF*. With that setting, the carriage return character (ASCII 13) and the line feed character (ASCII 10) will automatically be appended to whatever sequence of characters we type into the terminal program. As a result, we will be limited to a 126 character serial string because the addition of the *CR* and *LF* characters will bring us to the maximum of 128 bytes that can be stored in the 20X2 scratchpad.

As we'll soon see, we're going to use the fact that the last value in every incoming serial string will be 10 (line feed) to determine when to stop processing the serial input data. Finally, we also need to enable the *local echo* feature of the terminal program. That way, we'll be able to compare the string that we send from the computer with the string that's echoed back from the 20X2.

The breadboard setup for Experiment 3 remains exactly the same, except that we need to replace the 20M2 processor of Experiments 1 and 2 with a 20X2 processor. The program we'll use in this experiment is *hserial20X2direct.bas*. Before you download and run the program on your 20X2 breadboard setup, let's

take a look at the new features of the program. As usual, the following numbered comments refer to the corresponding numbers along the right edge of the program listing:

1. The *hserinptr* parameter is configured as follows: background receive to scratchpad, true serial output, true serial input, *hserout* enabled, *hserin* enabled. (See Figure 4 for the specific settings.)

2. The *hserinflag* is tested once in each iteration of the main *do/loop*. In larger more complex programs, you might want to test it more often.

3. The purpose of this *pause* statement is to allow enough time for all the data to be received before beginning to process it — even if the data length is the maximum of 126 characters, plus *CR* and *LF*. At the rate of 115200 baud, the delay is not necessary; the data is automatically received much faster than we can process it. However, at slower baud rates, the delay does prevent serial I/O errors.

4. Before entering the *do/loop*, we need to reset the scratchpad pointer to 0, so that the *do/loop* begins reading data at scratchpad location 0.

5. In this form of the *do/loop*, the value of *chr* is tested after it has been processed in the loop. As a result, when the last character in the serial string (which is always 10 [line feed]) has been processed, we exit the loop.

6. Before returning from the *GetNewData* subroutine, we need to reset the *hserinptr* to 0, so that storage of the next serial input data will begin at location 0 in the scratchpad.

7. We also need to reset the *hserinflag* to 0. If we didn't do that, the subroutine would execute again as soon as we returned from the subroutine (because *hserinflag* would still be 1), even if no new serial data had been received.

At this point, we're ready to test the *hserial20X2direct.bas* program.



Make sure your terminal software is set up as described above, and that you have replaced the 20M2 with a 20X2 processor. Download the program to your 20X2 setup, and type some strings (each terminated by pressing the *enter* or *return* key on your computer). You should see each string you enter displayed on two separate lines in your terminal program window.

The first line is produced by the terminal's "local echo," and the second line has been echoed back to the terminal program by the 20X2 processor. If your results are different, you will need to troubleshoot your breadboard setup and/or the configuration of your terminal program.

## Experiment 4: Responding More Quickly

If you spent some time experimenting with different input strings in Experiment 3, I'm sure you noticed a bit of a problem: Sometimes there was a second or two delay before the 20X2 echoed the received serial data. If you analyze the program, you will probably figure out the reason for the variable delays in the echoed data.

**Hint:** *The problem is definitely not caused by a delayed reception of the hardware serial data. That process happens so fast that it seems instantaneous.*

If you placed the blame on the `wait 2` statement in the main `do/loop`, you're absolutely right. That statement is probably executing for 99.9% of the time in every iteration of the main `do/loop`, so it's likely that many of the serial inputs will be received in the background fairly early in the two-second delay.

Whenever that happens, the `wait 2` statement has to time-out before the `hserinflag` is checked, which results in the noticeable delays in processing the echo back to the

terminal program. For most projects, the delay in response isn't a major problem, but there certainly are projects in which a serial command string from the PC requires a response in far less than two seconds.

One solution to this problem would be to divide a long pause into several shorter pauses, and intersperse them with the `if/then` statement that tests the `serinflag`. For example, in the `hserial20X2direct.bas` program, we could replace the `wait 2` statement and the `if/then` statement in the main `do/loop` with the following code snippet:

```
for index = 1 to 200
  pause 10
  if hserinflag = 1 then
    gosub GetNewData
  endif
next index
```

The above code snippet checks for serial input every 10 mS while it's producing the (approximately) two second delay for toggling the LED. As a result, the echoed serial data will be sent to the terminal program with a maximum response time of 10 mS, which is about 200 times faster than the original program. In order to carry out Experiment 4 to test this assertion, edit the `hserial20X2direct.bas` program as discussed above, save it as `hserial20X2fast.bas`, and download it to your breadboard setup. The improvement in the response time should be obvious.

There's another way to speed up the response time even more dramatically, but we've run out of space this month, so we'll save that one for our next Primer installment when we continue our exploration of PICAXE PC serial communication.

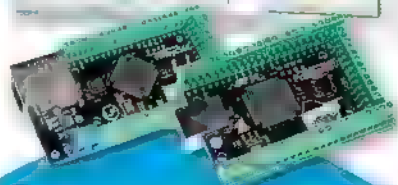
If you want to get the jump on things, you may want to read through the documentation for the `setinflags` command, and see if you can develop an interrupt-based solution to the response problem.

In the meantime, have fun ... **NV**

# Imagine this...

- a small and inexpensive
- USB or ethernet device
- with a rich set of features
- including PLC functionality and 3-axis stepper motor controller
- all accessible via free .NET/ActiveX or C++ library
- cross-platform
- configurable with free software

FORKEY



# Or this...

- all in one: Oscilloscope, Data Recorder, Logic analyzer, Analog and digital signal generator
- smallest USB 2.0 portable 1MS/s oscilloscope
- data acquisition of analog and digital signals
- data recording
- export to CSV, XLS, PDF and HTML
- simple usage of advanced features
- examples for C++, VB, Delphi and LabView
- free software and updates

PoScope Mega 1+





# Smartwatches Need Communications Too

## Apple's new Watch lives off of its wireless links.

**T**he new Apple Watch is the extreme example of a whole new category of consumer wearable products called smartwatches. You wear them like a standard wrist watch but they do so much more than just tell time. They are, in fact, simply extensions of your smartphone. Your smartphone is the computer; the smartwatch is a peripheral. This peripheral will let you do most of your smartphone functions from your wrist. It means not having to take your smartphone out of your pocket or purse to answer a call, text, or email. (We all know how inconvenient that can be.) What makes all of this possible are multiple wireless links between the watch and phone or other devices. Here is a brief look at how this works.

### The Apple Watch

The Apple Watch is a piece of work; refer to **Figure 1**. There is a lot crammed into that small package that comes in two sizes: 38 x 32 x 12 mm; and 42 x 42 x 12 mm. It has a full color touch screen that is about 1.5 inches diagonal in size. The technology is AMOLED, and the Retina resolution is 272 x 340 pixels for the smaller version or 312 x 390 pixels for the larger size. Of course, it has its own processor: the Apple S1 (whatever that is). There is 256 MB of RAM and 8 GB of other storage.

The Watch is designed to work with an iPhone. It will connect with iPhones 5, 5C, 5S, 6, and 6 Plus. Be sure you have the latest version of the operating system iOS8+.

As for I/O devices, the Watch has the following sensors for input: accelerometer, gyroscope, and a heart rate monitor. The touch screen also counts as an input, and a microphone is included. As for outputs, there is the AMOLED screen and a Haptics motor to vibrate a silent notification. A small speaker is included. This represents quite a bit of capability but remember, the Watch relies mainly on the iPhone for its function. However, you can still do quite a bit without touching the phone itself.

Some of the notifications you can get on the Watch are for an incoming call or missed call, email, text, calendar reminder, or alarm. Notification can be by speaker, screen, or vibration (see **Figure 2**). You can actually take a call by listening to and talking to your wrist if that affectation does not bother you. You can also use the Siri function. This is the voice query and intelligent

response feature of the iPhone. (Did I mention this smartwatch also tells time?) You can change the look of the watch on the screen, and the time is accurate to within  $\pm 50$  milliseconds. Not bad.

One key feature of the Watch is the wide availability of apps. The Watch is said to come with some already built in, like Facebook, Twitter, Uber, Instagram, plus some others like fitness apps using the heart rate monitor and GPS with the pedometer feature. The Apple website lists a whole slew of different apps, and they are counting on many more to come from third-party developers.

### The Wireless Ways of the Watch

Peripheral devices are typically tethered to a computer by a cable. Cables are so yesterday, ugly, and



■ **FIGURE 1.** The Apple Watch is a wireless accessory to an iPhone and comes in dozens of styles and price ranges.



Post comments on this article and find any associated files and/or downloads at [www.nutsvolts.com/index.php?/magazine/article/July2015\\_OpenCommunication](http://www.nutsvolts.com/index.php?/magazine/article/July2015_OpenCommunication).



■ **FIGURE 2.** Here are three common Apple Watch functions other than time keeping. From left to right: text messages, phone calls, and email.

inconvenient, though. That is why wireless technology is the key to the sophistication of the Apple Watch. The Watch is loaded with wireless. According to Apple, the Watch includes Bluetooth, Wi-Fi, NFC, and wireless charging. It is still unclear how some of these technologies are used and for what features.

I suspect Bluetooth is the main wireless link for notifications and calls. The Watch uses the latest Bluetooth 4.0 that includes the low energy version that conserves battery power. Also called Bluetooth Smart, it works great for audio and low speed texting. It works with Bluetooth headsets.

The Wi-Fi the Watch contains is specifically 802.11b/g/n. It can link to the iPhone when the watch goes out of range of the Bluetooth connection – probably by way of Wi-Fi's Direct Connect feature that lets Wi-Fi enabled devices talk to one another directly without having to pass through a router. The Watch can presumably also connect to a home router, but it is not known what function this serves.

Another wireless feature is NFC, or near field communications. This is the short range technology used

for Apple's new Pay system. NFC operates at 13.56 MHz, and can run at speeds from 106 kbps to 424 kbps over a range up to about 20 cm. Apple Pay is a way for you to charge purchases to your credit card by simply tapping your iPhone on a pay terminal or just wave your phone near the terminal. You can also do this with the Watch for convenience.

Finally, an unexpected feature is wireless battery charging. While the Watch is said to run up to 18 hours on its battery, it is one more item you have to charge each day. There is no direct electrical connection between the charger and the Watch. The charger is applied to the back of the Watch and it inductively connects to a coil inside. The coils in the Watch and charger form the primary and secondary windings of a transformer to transfer the power.

As I said earlier, the Watch has a lot of technology packed into it. Luckily, wireless chips are pretty small so aren't much of a problem. However, I wonder where they put the antennas. I suppose the Bluetooth and Wi-Fi chips can share a common antenna since both technologies work on the 2.4 GHz band. The 13.56

MHz NFC chip needs a fairly large coil for an antenna ... it must be in there somewhere.

As for GPS, there is no receiver in the Watch. You can use the GPS and maps on the iPhone and the maps will display on the Watch screen.

### The Future of the Smartwatch

The Apple Watch first became available in April. It is hard to determine just yet how well it will do. While the iPhone or any smartphone is basically a necessity today, the Watch is an accessory. It is a totally cool accessory with maxed out technology. But it is expensive. There are dozens of versions of the Watch. The cheap low-end model sells for a wallet-thinning \$349. The high-end gold plated models are said to cost as much as \$17,000. A typical model will probably set you back \$500 or \$600 – the cost of the phone itself.

Smartwatches have not sold well over the past several years they have been available. Most are simpler and cheaper than the Apple Watch, and most are used for health and fitness tracking. The Apple Watch is a whole new ballgame in the emerging field of wearables. Are you going to get one? Time will tell. **NV**



# NEW PRODUCTS

- HARDWARE
- SOFTWARE
- GADGETS
- TOOLS

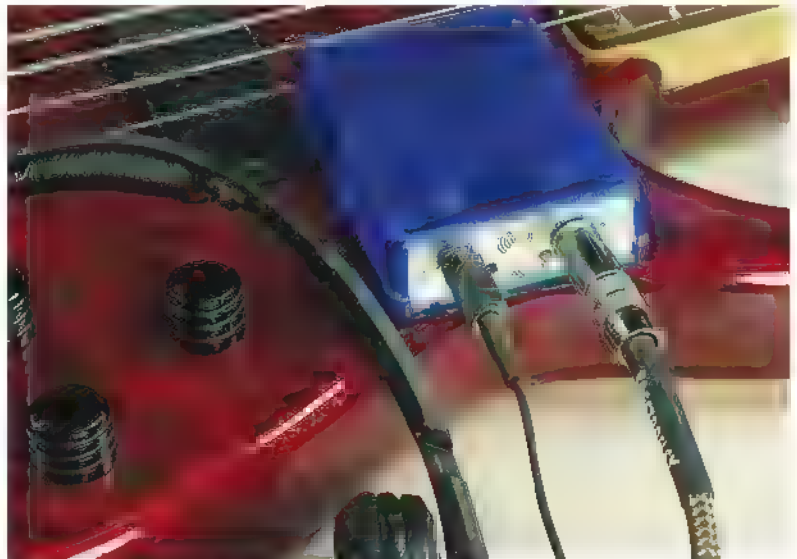
## DIY POCKET BASS AMP KIT

**B**oxed Kit Amps introduces their newest DIY electronics kit: the SoloB Pocket Bass Amp Kit. The completed amp is battery powered and will fit in your back pocket, allowing you to practice bass anywhere. It can be used with either headphones or an external speaker (a powered sub-woofer is recommended) and – with the proper cabling – it can be used as a DI (direct injection) adapter; for example, to plug directly into your computer for recording.

“The SoloB amp is a reengineered version of our guitar amp, the So oG,” said designer, Ken Knowles. “Not only is the bass in a lower register, but the signal level has a much wider dynamic range. You can use a guitar amp with the bass, but it will sound much better with an amp specifically design for the bass.”

The SoloB uses a genuine JRC386D op-amp just like the original “Smokey” amp. A JFET input buffer gives it a clean powerful tube-like sound. The amp is responsive to the bass’ volume and tone knobs. A standard nine volt battery lasts for many hours. Plugging in a patch cord automatically turns the amp on. A green LED on the faceplate indicates power; a red LED acts as a clipping indicator.

The kit provides a learn-to-solder project or a quick fun build for a more experienced maker. It is suitable for



anyone qualified to handle a soldering iron. There are only through-hole components and the only wiring is to the battery holder.

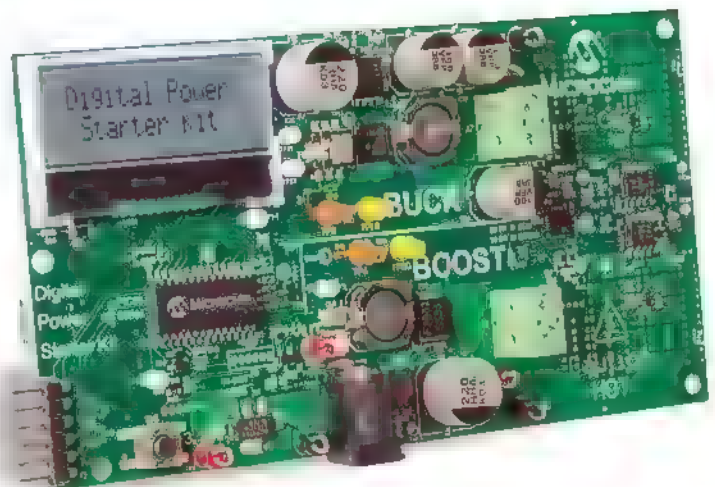
The kit comes complete with circuit board, components, case, wiring, headphone adapter, and battery. There is nothing more necessary to buy for a complete working project. The SoloB retails for \$25.50, which includes US shipping.

For more information, contact:  
**Boxed Kit Amps**  
[www.boxedkitamps.com](http://www.boxedkitamps.com)

## dsPIC33EP FAMILY OPTIMIZED FOR DIGITAL POWER APPS

**M**icrochip Technology, Inc., is introducing their new 14-member dsPIC33EP “GS” family of Digital Signal Controllers (DSCs). The dsPIC33EP GS family delivers the performance needed to implement more sophisticated non-linear, predictive, and adaptive control algorithms at higher switching frequencies. These advanced algorithms enable power supply designs that are more energy efficient and have better power supply specifications.

Higher switching frequencies enable the development of physically smaller power supplies that offer higher densities and lower costs. Compared with the previous generation of DSCs, the new dsPIC33EP GS devices provide less than half the latency when used in a three-

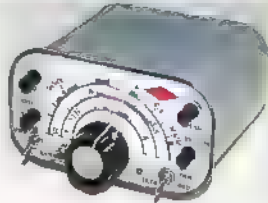


# NATIONAL RF, INC.

TYPE **75-NS-3**

## MINI HF RADIO

The 75-NS-3 covers 3.5 to nearly 11 MHz and is offered as a semi-kit.



**CUSTOMER PUTS IT IN A MEAT CAN!!**  
Visit [www.NationalRF.com](http://www.NationalRF.com) for this and other Radio Products!  
Office: 858-565-1319

Recycling & Re-marketing High Technology

# WEIRD STUFF WAREHOUSE

Software, Computers, Electronics, Equipment, Bookshelves

WE BUY/SELL EXCESS & OBSOLETE INVENTORIES  
FREE COMPUTER AND ELECTRONIC RECYCLING  
GIANT 10,000 SQ. FT. AS-IS SECTION

384 W. Caribbean Dr.  
Sunnyvale, CA 94089  
Mon-Sat: 9:30-6:00 Sun: 11:00-5:00  
(408)743-5650 Store x324  
[WWW.WEIRDSTUFF.COM](http://WWW.WEIRDSTUFF.COM)

[www.boxedkitamps.com](http://www.boxedkitamps.com)

Build something you'll enjoy every day.  
DIY HiFi Audio kits.

# boxed

NOT A MESS

In the Nuts & Volts Webstore NOW!

# NUTS AND VOLTS

## NV BOOK SPECIALS

Programming the Raspberry Pi  
Getting Started with Python

Simon Monk

Raspberry Pi  
Three Book  
Combo

Raspberry Pi  
user Guide

Raspberry Pi  
Projects for the  
EVIL GENIUS

**Only \$48.95**

Plus  
FREE Priority Mail Shipping  
US Only

To order call 800 783-4624 or visit  
<http://store.nutsvolts.com>  
*Limited time offer*

**ROBOT POWER** Extreme Motor Speed Control!

MegaMoto GT- Mega Motor Control for Arduino™

- 6V 10V 35A / 50A peak
- Single H-bridge or dual half
- Current and Temp protected
- Jumper select Enable & PWM
- Robust power terminals

The VYPER™

- 8V-42V - 125A / 200A peak!!
- Single H-bridge
- RC and Pot control
- Overload protected
- For your B/G bots

[www.robotpower.com](http://www.robotpower.com)  
Phone: 360-515-0691 • [sales@robotpower.com](mailto:sales@robotpower.com)

**NKG electronics**

# MDE8051 Trainer

by Digilent

[NKGelectronics.com/MDE8051](http://NKGelectronics.com/MDE8051)

The MDE8051 trainer kit includes:  
supply, serial cable.

Purchase Orders are accepted from Educational Institutions,  
US Government and Research Centers.

[www.Primecell.com](http://www.Primecell.com)

## Battery rebuilding service

Dead Batteries? Don't toss them.  
Send them to us - our rebuilds are better than original specifications.

**Tools**

Hilti Skil  
Milwaukee  
Panasonic  
B&D DeWalt  
Makita All  
2-36 Volts

**Electronics**

Bar Code  
Scanners  
Surveying  
Printers  
Laptops  
Photography

**Radios**

APELCO  
UNIDEN  
GE ICOM  
KENWOOD  
MOTOROLA  
MIDLAND  
MAXON  
YAESU  
ALINCO

Visit [www.primecell.com](http://www.primecell.com) for important details  
24 Hr Secure recorder tel fax (814) 623 7000  
Quotes email: [info@primecell.com](mailto:info@primecell.com)  
Cunard Assoc. Inc. 9343 US RT 220 Bedford PA 15522

**Images Scientific Instruments Inc.**

## PIC Basic Project Board

### 16F88 PIC Basic Project Board Features

LCD Display - Backlight & contrast control  
2 A/D Channels  
4 Digital I/O pins  
5V and 9V operation  
Free Student version of PICBasic Pro & Microcode Studio

Do More Projects.

Frequency Meter	Sensor Reader
Pulse Generator	Eclipse Timer
Radiation Pulse Counter	Volt Meter
Humidity Sensor	Toxic Gas Sensor

[www.imagesco.com/microcontroller/index.html](http://www.imagesco.com/microcontroller/index.html)

pole/three-zero compensator, and consume up to 80% less power in any application.

This new dsPIC33EP family includes advanced features such as Live Update Flash capability, which is especially helpful for high availability or "always on" systems. Live Update can be used to change the firmware of an operating power supply — including the active compensator calculation code — while maintaining continuous regulation. Variants from this new digital-power-optimized DSC family are available in a small 4 x 4 mm UQFN package for space-constrained designs.

Other key features of this family include up to five 12-bit ADCs with as many as 22 ADC inputs, providing total

throughput of 16 Mega samples per second (Msps) with a 300 ns ADC latency. The dsPIC33EP GS devices include 12-bit DACs for each of the four analog comparators for higher precision designs.

The two on-chip programmable gain amplifiers can be used for current sensing and other precision measurements. Including these advanced analog amplifiers on the device reduces the number of external components required, thereby saving costs and board space. These features — combined with the overall high performance of the GS family — make it well-suited for a wide range of applications. For example: computer and

*Continued on page 77*



# MODIFY A KILL-A-WATT EZ POWER METER FOR

By Kevin J. O'Connor

# LOW VOLTAGE OPERATION

Post comments on this article and find any associated files and/or downloads at [www.nutsvolts.com/index.php?/magazine/article/july2015\\_OConnor](http://www.nutsvolts.com/index.php?/magazine/article/july2015_OConnor).



Kill-A-Watt EZ.

The Kill-A-Watt EZ is a very low priced digital AC power meter with logging features. It is very useful around the home and shop for measuring energy use and power consumption of 120 VAC devices. It will measure current, voltage, real and apparent power, frequency, and power-factor (PF). For most appliance measurements, one is only interested in real power (what you pay for!) and possibly PF. However, this device could be used for a lot more in the shop and/or on the bench. I do a lot of work with power transformers, and two very common tests to characterize them are open-circuit (OC) and short-circuit (SC) tests. From these tests, the resistive and reactive transformer parameters are determined.

While the transformer testing process is outside the scope of this article, the conditions for OC and SC testing bare directly. OC testing looks at full voltage, low input current, and no-load current (open secondary) conditions. SC testing looks at low voltage, high input current, and no-load voltage (shorted secondary) conditions. The Kill-A-Watt has the potential to provide all of the measurement parameters needed for both OC and SC testing, except for one problem. It is designed to operate at a full AC voltage (nominally 120V). If this voltage drops below about 60 VAC (say via a variac autotransformer **Figure 1**), the simple internal DC supply for the microcontroller begins to fail, and the power meter displays erroneous results.



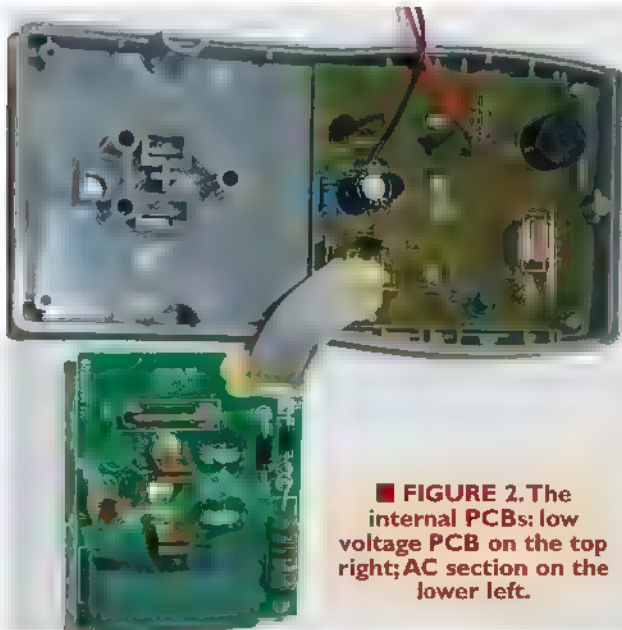
■ **FIGURE 1.** A typical variac transformer. Photo courtesy of allelectronics.com.

OC tests work fine as full voltage is used, but for SC testing only enough voltage is used to drive the transformer to the rated input current. This may require only 10%-20% of full voltage, and the stock power meter will not work with such a low input voltage.

This article describes a simple modification that safely powers the meter for accurate operation in this low voltage regime.

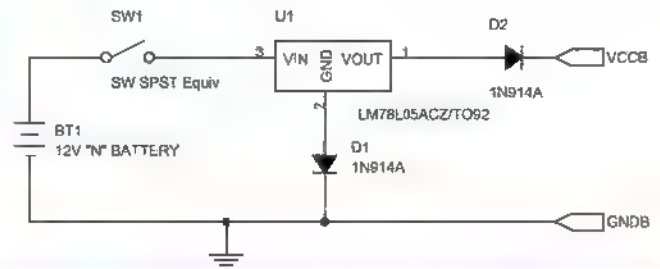
## Open the Meter

The lead photo on the previous page shows the Kill-A-Watt EZ model 4460 front. Notice it is not plugged in! Nothing inside is isolated. Three screws on the back will separate the covers and provide access to the two internal



■ **FIGURE 2.** The internal PCBs: low voltage PCB on the top right; AC section on the lower left.

**Kill-A-Watt Low Voltage Operation Mod**  
 SW1 Mouser 612-FBH2UEENAGX /612-1R-BK  
 BatHolder Mouser 534-2471  
 U1 Mouser 512-LM78L05ACZ  
 BT1 A23 12V  
 D1,2 1N914A or equiv.



■ **FIGURE 3.** Schematic and parts for modification circuit.

circuit boards. **Figure 2** shows the inside of the top with the AC board separated.

We will make no changes to this printed circuit board (PCB). The controller board is to the right and does not need to be removed from the case. The microcontroller (not visible from the top side) requires a standard 5 VDC power supply. Derived in part from a dropping resistor on the AC PCB, it fails when the input AC is too low. Though the internal supply uses a 78L05 voltage regulator already, we will add another regulated, switchable, and battery powered supply for the low AC region of operation.

The schematic of **Figure 3** shows the simple LM78L05-based supply which is powered through a pushbutton from a 12V "N" size cell. The output diode (D2) provides isolation and the ground diode (D1) compensates for the output diode voltage drop. The entire circuit fits on a small perf-board so that the only "permanent" modification is drilling a hole in the top of the back cover for the pushbutton actuator.

## Make the Mods

In **Figure 4**, there are two wires visible (red and black) connected to the Vcc (+5V) and digital ground for the controller. Insure that your K-A-W PCB matches the layout



■ **FIGURE 4.** Power connections.





■ FIGURE 5. Battery holder mount.

in the picture. Specifically, jumper JP6 = VCC and the top end of R33 = digital ground.

Figure 5 shows the N size battery holder epoxy-mounted into the top of the back cover. Note the orientation and fit.

## The Circuit Board

The new power supply board is only slightly larger than the latching pushbutton. The exact size will depend on the chosen button. Any latching *plastic insulated* tip single-pole single-throw (SPST) equivalent will do. **Repeat: Don't use a metal switch!** Don't drill the button hole until the board is completed.

Then, cut the hole to match the fitted board. Note the output diode (D2) on the right and the offset diode (D1) under the 78L05 regulator (Figures 6a/6b). This board is assembled with push-pins, but any hand-wired method will work.

Figure 7 shows the circuit board in position in the upper right corner of the plastic back. I held the board in position with a dab of hot glue.

There is sufficient room in the case for all these components, but upon reassembly, insure that the loose

**Tip:** *In assembling the PCB, I found it easier to glue in the PCB before attaching the connecting wires to the rest of the K-A-W.*

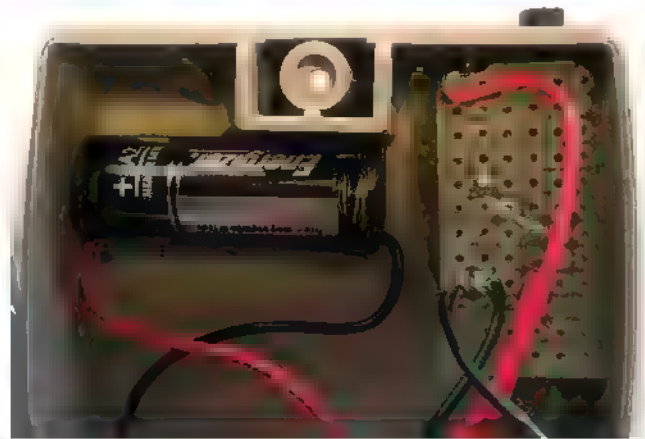
wires are positioned above the controller PCB and do not loop down into the AC board area. Final layout for the project is shown in Figure 8.

## Operation

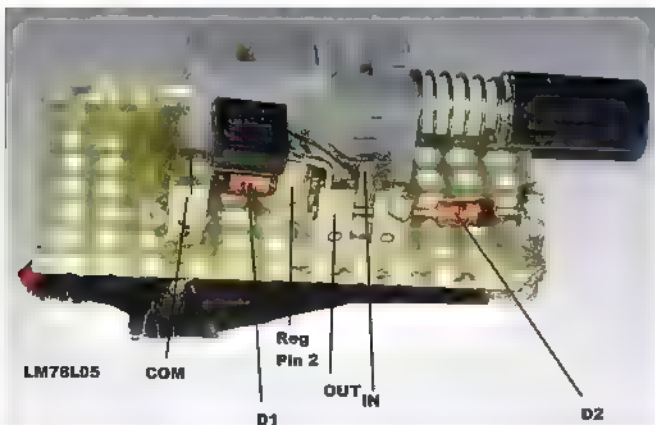
The switch specified in the BOM is a latching type push-on/push-off button. The K-A-W internal accuracy falls around 40 VAC, so the circuit should be engaged when the incoming line voltage drops below about 50 VAC. Since I am usually working in the 10-50 VAC range, I leave the battery engaged for the duration of my measurements. But do not forget to disengage the battery when done.

Since the response of the K-A-W to the battery is almost instantaneous, one could use a momentary button instead, and inherently save on battery life. However, one may also need their fingers elsewhere when making measurements.

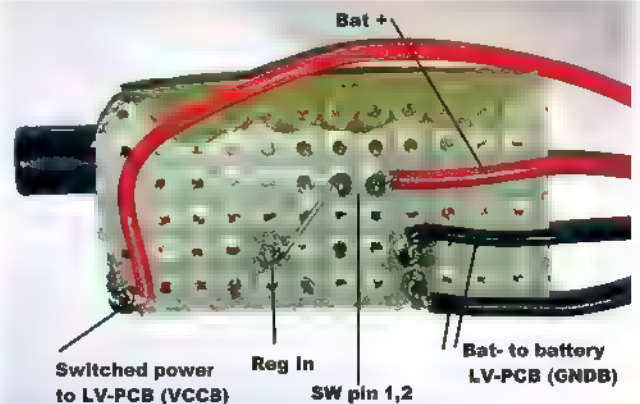
The K-A-W draws 7 mA at 5 VDC (from LM78L05) at the digital PCB, and 9.7 mA from the 12V battery. A 2-3 mA bias is typical for the LM78L05. Note that on my K-A-W units, the internal power supply sits at 4.95V even at 120 VAC in, so the LM78L05 circuit at 5V is always supplying the PCB power if engaged. Some readers may be tempted to



■ FIGURE 7. Installed battery and PCB.

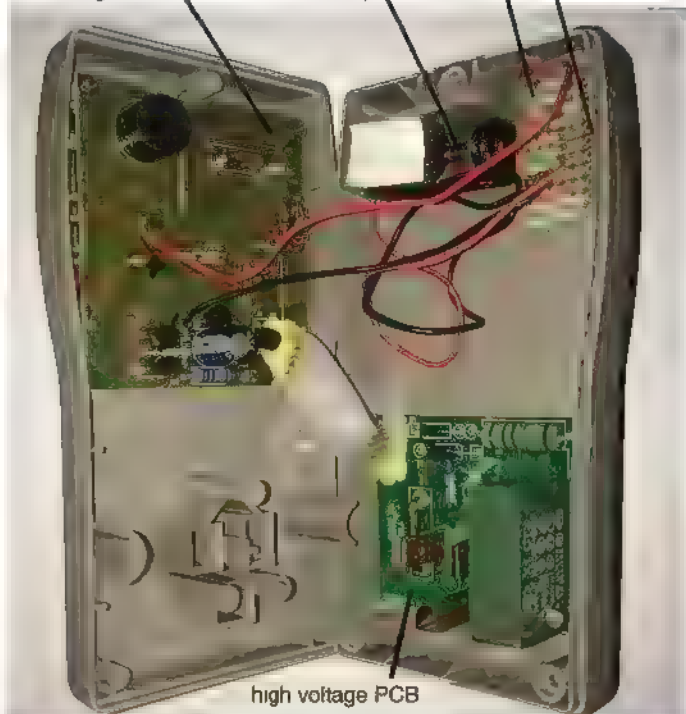


■ FIGURE 6a. Assembled switch PCB.



■ FIGURE 6b. Underside of the assembled circuit board.

Carefully epoxy (or hot glue) this edge to case. NO glue into PBSW!  
 low voltage PCB      12V battery      new PCB



■ FIGURE 8. Final assembly layout before reassembly.

**Tip:** Don't forget to turn the circuit off when not in use. The 78L05 idles at 3-5 mA, so the A23 won't last forever. There are other 5V regs with lower idle and lower drop-out.

use a lower bias current regulator. Remember, however, that the bias current in the regulator sets the output voltage compensation drop at D1.

Skipping the math, if the currents in D1/D2 differ by a factor of 10, their voltage drops will differ by ~100 mV. Be aware, as that will lower the output voltage to maybe 4.9V and may affect the K-A-W performance.

## Caveats

I have had these Kill-A-Watt EZ units for a while. It is possible that the current production from P3International/Prodigit is different, but access to the +5/gnd is all that is needed. Careful tracing will locate them.

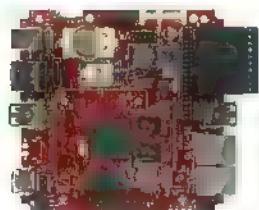
**A caution to reiterate:** There is no AC line isolation in this meter! All digital circuits are floating at the line voltage, so don't plug in the meter to make internal measurements. Use an isolation transformer instead. Don't make modifications that protrude beyond the case if they are conductive; hence, the use of an insulated plastic pushbutton. **NV**

## SUPERIOR EMBEDDED SOLUTIONS



DESIGN YOUR SOLUTION TODAY  
 CALL (480) 837-5200

[www.embeddedARM.com](http://www.embeddedARM.com)



### TS-7970 Single Board Computer

Industrial High Performance i.MX6 Computer with Wireless Connectivity and Dual GbEth

- 1 GHz Solo or Quad Core Freescale i.MX6 ARM CPU
- 2 GB RAM, 4 GB eMMC Flash
- WiFi and Bluetooth Module
- 2x Gigabit Ethernet, 4x USB
- HDMI, LVDS, & Audio In/Out
- Linux, Android, QNX, Windows



Module Starting At  
**\$89** (Qty 100)

Starting at  
**\$169**  
 Qty 100  
**\$214**  
 Qty 1



### TS-TPC-7990 Touch Panel PC

7" High End i.MX6 Mountable Panel PC with Dev Tools Such as Debian GNU and QTcreator

- 7 Inch Touch Panel PC Powered by 1 GHz i.MX6 ARM CPU
- Resistive and Capacitive Screens
- 10 Inch Screen Available
- Linux, Android, QNX, & Windows
- QTcreator, GTK, DirectFB, and More
- Yocto, Debian, Ubuntu Distro Support



Enclosed TPCs  
 Also Available

Starting at  
**\$299**  
 Qty 100  
**\$342**  
 Qty 1



100% never discontinued a product in 30 years



rugged systems that are built to endure



support every step of the way with open source vision



unique embedded solutions add value for our customers



# Beyond the Arduino

5

## Please DO Interrupt Me

Sometimes interruptions can be a good thing. This month, we look at how to handle interrupts in your Atmel AVR embedded projects, and how they can make your projects more efficient and easier to develop.

### Sorry for the Interruption

We exist in a world where we try to shy away from interruptions — we teach our children that it's rude to interrupt; we become frustrated when colleagues interrupt us and disrupt our thought processes. In short: Interruptions are bad.

Many years ago, I was lucky enough to spend a couple of years working as a guide at a luxury safari lodge in South Africa, where interruptions were an integral part of life. A herd of zebra would be “interrupted” by the sudden arrival of a pride of lion, and my conversations with my guests would be “interrupted” by a rustling in a nearby bush. At dinner around a roaring fire, our chef (who was from a local rural community) would arrive and loudly declare “Sorry for interruption” as he prepared to announce the menu for the evening — an interruption that our rumbling stomachs always welcomed!

The point I'm trying to make is that sometimes interruptions are useful and require your immediate attention — particularly when we're dealing with the world of microcontrollers. This month, we'll be interrupting our microcontroller ... and without any apology at all!

### Just What is an Interrupt?

An interrupt is, simply, a way to pause what your microcontroller is busy doing (or not doing, as is likely the case) and focus it on a task that needs attention. If your

background is similar to mine (originally software development), you could (kind of) think of an interrupt as the hardware equivalent of an event in event-driven programming models.

When an interrupt fires, the MCU drops what it's busy with and dashes off to process the urgent task that the interrupt triggered — this task is contained in a function called an *ISR* (Interrupt Service Routine). When the ISR has completed execution, the MCU picks up where it left off originally.

As an example, you might want to use an interrupt to handle an emergency button that shuts a motor down. You don't want to wait until a sequence of flashing LEDs have completed their choreography — you want to action the button-press immediately before you start losing fingers in machinery! Once your fingers are safe, then the LEDs can carry on with their blinking.

You may have used interrupts when working with Arduino boards in the past, through the Arduino IDE's (integrated development environment) built-in interrupt management functions: *attachInterrupt()*, *detachInterrupt()*, *interrupts()*, and *noInterrupts()*. These functions control the interrupts that we most commonly think of when we hear the term: those triggered by I/O pins. However, these are not the only types of interrupt. Let's take a quick look at the other types we can access on our microcontroller now that we're moving beyond the Arduino.

#### I/O Interrupts

I/O interrupts are those triggered by — not surprisingly — the I/O pins on the microcontroller. There are two types of I/O interrupts on the ATmega328P: external interrupts and pin change interrupts. This might start to sound rather complex, but bear with me — it really isn't.

**Pin Change Interrupts** are the simplest: Each I/O pin on the microcontroller can trigger an interrupt when it toggles (i.e., goes from low to high or from high to low). If you refer to the pin configuration diagrams, you'll see that each pin has a **PCINTxx** associated with it. **Figure 1** highlights these.

These **PCINT** numbers are what we use to configure and handle the interrupts. Although each pin can trigger an interrupt, the pins are grouped into three “banks,” with

each bank having a single interrupt associated with it. Within each bank, we can set which individual pins trigger an interrupt. However, we can't (using registers) easily work out which pin within a bank triggered an interrupt. To do that, we'd need to do some coding that compared actual pin values to stored values.

**External Interrupts** have additional functionality but are only available on pins PD2 and PD3. They are labelled **INT0** and **INT1** (refer to **Figure 1**), and can be configured to trigger more complex interrupts. They can trigger on a rising edge (a change on the pin from a low to a high state), a falling edge (change from high to low state), a pin toggle, and on a low signal (the interrupt will continue to trigger while the pin is low).

Both of these interrupts have their own dedicated handlers.

### Time-Based Interrupts

We haven't yet touched on timers, and they don't form an obvious part of the Arduino IDE, so may not be familiar to you. However, we will delve into them in a future article. In brief, your ATmega328P has a number of internal timers that tick away every clock cycle, and are useful for measuring elapsed time. These timers can be configured to generate an interrupt when they reach certain values – think of them as tiny alarm clocks that trigger an alarm every x milliseconds.

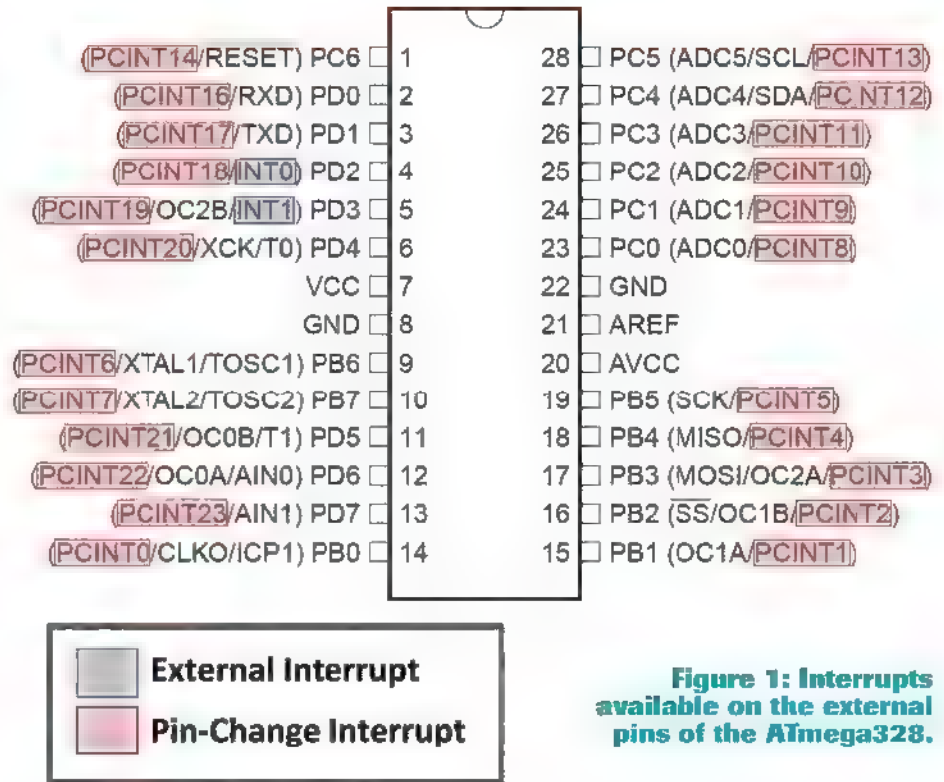
In addition, there is a special kind of timer called – interestingly – a watchdog timer. Again, we won't have time to go into watchdog timers now, but suffice it to say they are also capable of generating interrupts.

When we deal with timers in a month or two, you'll see that the interrupts are fundamental to using them.

### Communication-Related Interrupts

Last month, we dealt with serial communications using the UART. You may remember when we transmitted a data frame that we had to wait in a loop for the data register to be empty before we could send the next byte. Using interrupts, we can rewrite this code so that we aren't waiting around for the buffer to empty – we can get the UART to *tell us* by triggering an interrupt.

In addition to this interrupt, we can also get the UART to tell us after a data frame has been transmitted, and when data has been received on the UART. I'm sure you



**Figure 1: Interrupts available on the external pins of the ATmega328.**

can see that these interrupts will remove the need for us to keep polling various registers. We can keep working, and (for example) only worry about reading incoming data once there is actually data coming in.

Similar interrupts exist for other serial modes and protocols such as SPI and I<sup>2</sup>C (TWI).

### Other Peripheral Interrupts

A number of other peripheral interrupts exist. The ADC (analog-to-digital converter) we looked at in the May 2015 article can generate interrupts when it has completed a conversion. Again, this means we don't need to sit in a loop waiting for the ADC; we can carry on with other tasks and then deal with the result when the conversion is complete. The analog comparator can also generate interrupts, as can the EEPROM module.

## Please Do Not Disturb

Before we start on a practical project, let's briefly look at why we'd want to hang a "Please Do Disturb" sign in our projects, as well as at a couple of cardinal rules.

### Tell Me Why

From the discussion that we've had so far, you've probably picked out a number of reasons why we'd want to use interrupts in our projects. Here are some of the



## Volatile Keyword

The *volatile* keyword is often misunderstood — I used to use it sporadically and without really knowing why. It was one of those things I never really got into the details of, but finally got around to researching and understanding.

As you know, our MCU has very little memory (compared to a PC) and runs very slowly (compared to a PC). As a result, optimization is an important requirement when writing code for embedded systems. Humans can write optimized code, but it takes a great deal of effort and often results in code that is not very readable or easy to maintain. So, the clever people who write compilers (and they are in the realms of the super-intelligent) have included a number of ways that the compiler can optimize code.

One of those optimization methods works by removing variables where they aren't needed. For example, we may use a variable to store a value, but we may never actually change the value of the variable — essentially using it as a way to make our code easier to read. The compiler picks this up and so instead of allocating memory for a variable, it simply substitutes the variable with a fixed value. The compiled code is now smaller and runs more efficiently.

There is a small problem with this, though. The compiler works on sections of the code and may not see whether a variable is being changed in an ISR. The result is that the compiler can “optimize out” the variable in the main routine by substituting it with a fixed value. Then, when the ISR changes that variable, the main routine is running off a fixed value and so does not “see” the change.

The *volatile* keyword prevents the compiler from optimizing that specific variable, therefore allowing the changes that the ISR makes to be seen in the main routine. To see this in action, pop over onto YouTube (see Resources) and see Atmel illustrate it themselves

reasons that I use interrupts in mine:

**Urgency:** I gave an example earlier of using interrupts on an emergency stop button to prevent machinery from removing parts of your body. There are, of course, many more less extreme reasons why you'd want to process inputs urgently; for example, creating a better user experience. A rotary encoder is a good case in point: You want your project to respond as the encoder clicks through each detent, or the user will become frustrated when interacting with your project and its erratic skipping and stuttering interface.

**Reduced Power Consumption:** With the Internet of Things (IoT) being such a hot topic (the *value* of IoT is a debate for another time!), power consumption of projects is gaining a focus. Microcontrollers have various levels of “sleep mode” which allow them to shut down in order to save power. However, something needs to wake them up again to do what they need to do, and it's up to an interrupt to do this — whether on an external pin, a timer, or incoming data.

**Simpler Modular Programming:** By using interrupts, you remove the need to continually poll GPIO (General Purpose I/O) pins in the main loop of your program. You can push some of the processing out to functions that only execute when the interrupt fires. You're also able to avoid the “wait” loops that we used when waiting for ADC conversions or UART transmissions to complete. In some cases, you can almost “set and forget.” For example, an LED flasher can be implemented by activating a timer and then taking care of the flashing completely in the interrupt handler.

### Tell Me What and What Not

There are a number of considerations when using interrupts. If you have used the simple I/O interrupts available through the Arduino IDE, you may already be familiar with these. We could easily launch into some fairly lengthy discussions on the best ways to use interrupts — but I'm sure you'd rather save that for another time!

**Keep it Short:** When an interrupt fires and control passes to the ISR, the ATmega328 disables *all* interrupts until the ISR has completed and control has returned to the previous routine. If you take too long handling this interrupt, then you could be missing other interrupts that fire. Therefore, never use delay loops or halt execution waiting for specific responses. This undermines one of the reasons for implementing interrupts: to have a system that can deal with time-critical events.

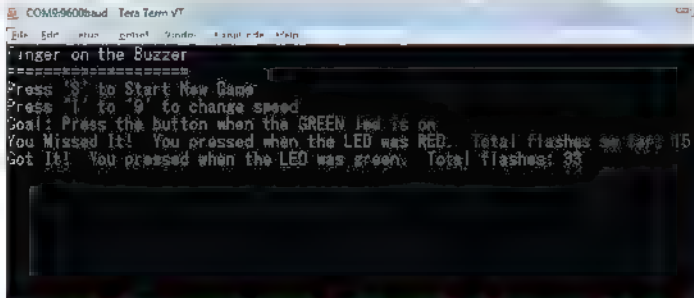
**Sharing Variables:** In order to keep your ISR as short as possible, you'll probably need to pass data back and forth between the ISR and your *main()* loop — for example, passing back a byte that was received on the UART. You may also want to look at implementing a system of flags, where the ISR sets a flag to indicate to the *main()* loop that it needs to take some course of action. As it wouldn't make sense to *RETURN* a value from an ISR (it is not ever explicitly called), you'll need to rely on globally declared variables as they are accessible in both the ISR and the main loop. One gotcha is that you need to use the *volatile* keyword when declaring the variables. This prevents the compiler from optimizing the variable out of the compiled code. (Refer to the **sidebar**.)

**Finite State Machines:** I won't go into the topic of state machines here, other than to mention that they can be useful in combination with interrupts. If you'd like to delve into state machines in more detail, I recommend a couple of hours on Google and a strong pot of coffee!

**Nested Interrupts:** You can re-enable interrupts from within the ISR in order to allow nested interrupts. Proceed with caution! It can get messy and out of control very quickly!

### Tell Me One Last Thing

Okay, the last thing before we get into a practical example: enabling, disabling, and clearing interrupts (I know that sounds like three things, but I'm going to cheat



**Figure 2: Screenshot of the terminal output from the "Fingers on the Buzzer" project.**

and group them into one!). To enable a specific interrupt, you need to do two things: set the Interrupt Enable bit in that specific interrupt's register, and then also make sure that the Global Interrupt Enable (GIE) bit is set. The GIE bit allows you to turn on (or off) all interrupts with a single statement.

Remember that all interrupts are disabled after any interrupt has fired? Well, the MCU controls that by unsetting the GIE bit at the start of the ISR and then re-setting it when the ISR exits. To easily set/unset the GIE bit, you can use two macros defined in the "interrupt.h" include file: *sei()* sets the GIE bit; and *cli()* clears (or unsets) the GIE bit. You'll see this all in action in a moment.

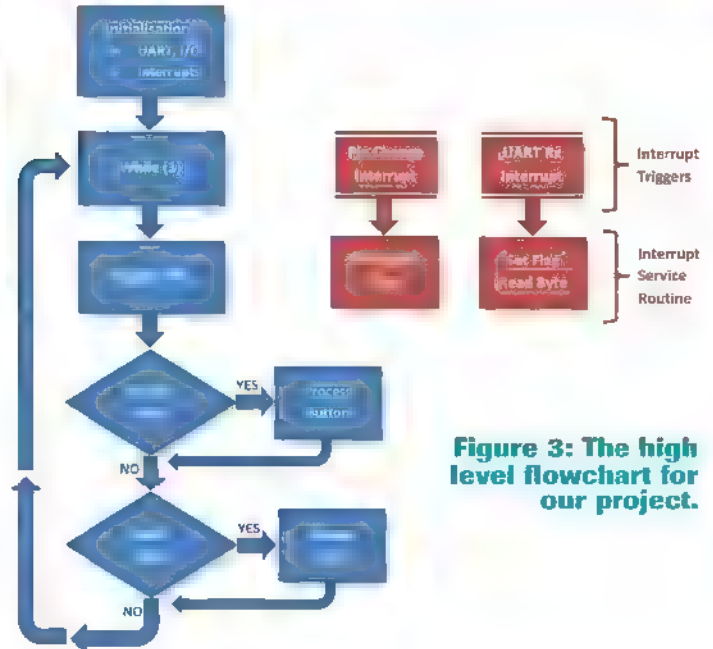
The last thing to consider is interrupt flags. Certain interrupts set an interrupt flag when they trigger. For example, the pin change interrupt sets the flag **PCIF0** (Pin Change Interrupt Flag 0) in the **PCIFR** (Pin Change Interrupt Flag) register. These flags are automatically cleared when the ISR exits, or can be cleared programmatically by writing a logic 1 to it. Why would we want to programmatically clear an interrupt flag? Surely, the ISR will have started processing the interrupt as soon as it fires.

Well, interrupts with flags are "stored" when interrupts are disabled, and then trigger once interrupts are re-enabled. Let's say that your program is executing inside the ISR — remember that global interrupts are disabled on entry to an ISR. If a pin-change interrupt occurs while global interrupts are disabled, the pin-change interrupt flag will be set. As the ISR exits, global interrupts are automatically re-enabled — and the pin-change interrupt that was flagged will then trigger. You may, therefore — depending on what task your project performs and what interrupts it handles — need to clear these flagged interrupts before exiting your ISR.

Now that we've dealt with that "one" last item, we can get into our project for this month's article.

## Finger on Your Buzzer

I had a number of ideas for a practical project

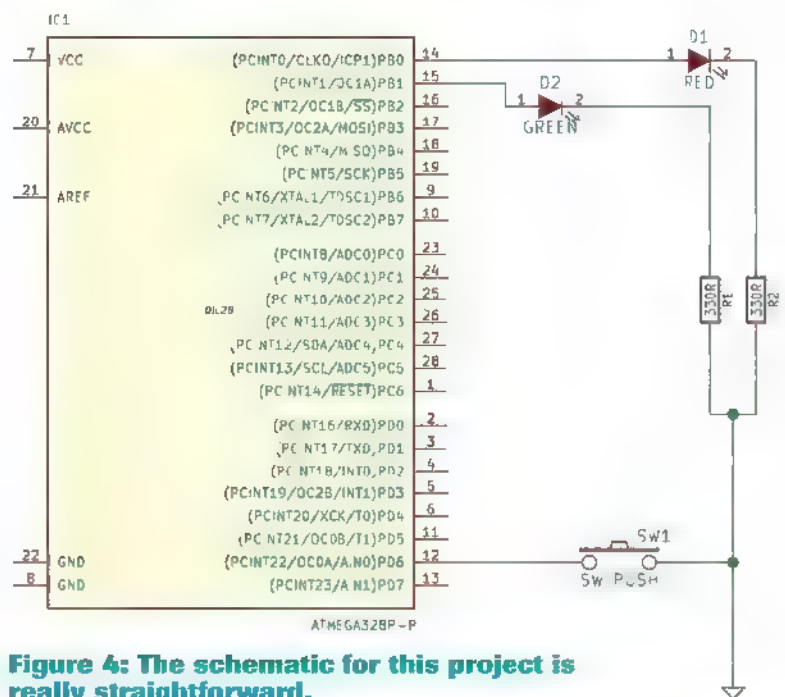


**Figure 3: The high level flowchart for our project.**

for this month's article. We could include interrupts in pretty much all of our projects so far in this series. In the end, I settled on one that covered two concepts: the immediacy of handling an interrupt; and the way in which an interrupt can remove the need to continually monitor an infrequently occurring event. So, we'll be creating a simple game that tests your reaction times — Put Your Fingers on Your Buzzers!

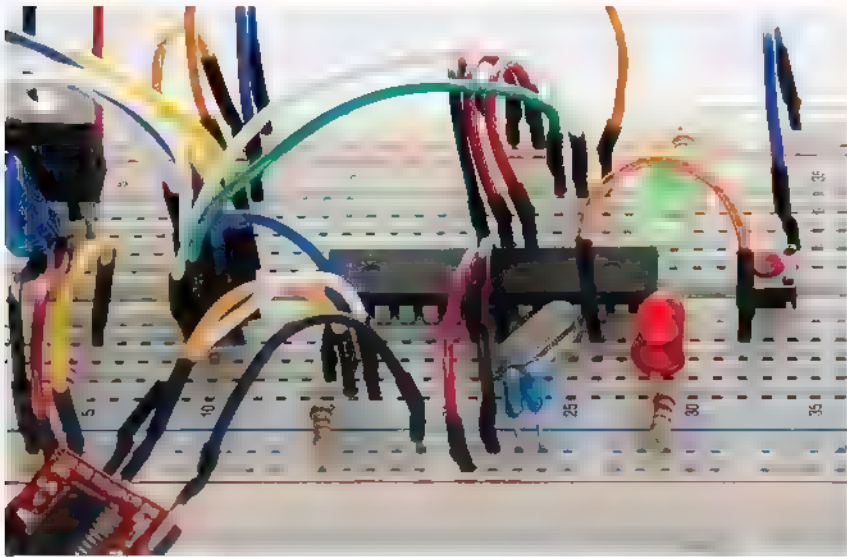
### The Game Play

The game is meant for two players competing for the



**Figure 4: The schematic for this project is really straightforward.**





**Figure 5: "Fingers on the Buzzer" on a Breadboard.**

fastest reaction times across nine levels. The breadboard has two LEDs — a red and a green — that flash alternately, and a single pushbutton. The goal is to hit the button while the green LED is lit, within the fewest number of

flashes. Each round starts with an LED count-down sequence before the LEDs start their alternating flashing. The levels are controlled through a terminal over the UART. Numbers "1" to "9" control the speed the LEDs flash at (with 9 being the slowest), and the letter "S" starts a new game. Additionally, the terminal displays the scores (Figure 2)

Take a look at the flowchart in Figure 3 for a summary of the program flow, and download the full source code from the article link. We'll go through excerpts of the code here to dig into the configuration and use of the interrupts.

The schematic and final breadboard layout are really straightforward, but I've included them here for good measure (Figures 4 and 5). Of course, you need to connect your USB-to-serial board to the Tx and Rx pins on your MCU.

```
int main(void)
{
    ===== CUT =====

    //Initialise Interrupt: Pin Change
    PCICR |= (1<<PCIE2);
    //Enable interrupts on PCINT "bank" 2
    PORTC.DR |= (1<<PCINT22);
    //Enable interrupts on PCINT "bank" 2, which is
    //PCINT22 (within "bank" 2)

    //Initialise Interrupt: UART Receive
    UCSRB |= (1<<RXCIE0);

    //Enable Interrupts
    sei();

    while(1)
    {
        ===== CUT =====

        /*****
        * Handle flags returned by Interrupt
        * Handle
        *****/

        if (flagButton == 1)
        //Button was pressed - the
        //flagButton variable is set in the ISR
        {
            processButtonPress();
        }

        if (flagUART == 1)
        //A character was received over the
        {
            processUART();
        }

        /*
        Finished handling interrupt f
        *****/
    }
}
```

**Listing 1. Excerpt of the interrupt-related project code.**

```

===== CUT =====

.....

the ISR (interrupt service routine,
* It is triggered on a pin-change for "bank"
)

*****/
ISR(PCINT2_vect)
{
    PCICR &= ~(1<<PCIE2);
    //Disable interrupts on PCINT "bank"
    //prevents interrupt on switch 1
    flagButton = 1;
    //Set the flag so main routine knows to
    handle UART event
}

.....

This is the ISR (interrupt service routine)
for UART RX
It is triggered when a byte is received
over the UART
.....
ISR(USART_RX_vect)
{
    uartReadByte = UDR0;
    //Read the received byte from the
    flagUART = 1;
    //Set the flag so main routine knows to
    //handle button pr
}
```

## Let's Interrupt Our Project

Our project uses two types of interrupts: a pin-change interrupt and a UART “data received” interrupt. The pin-change interrupt is driven by the pushbutton, and demonstrates the use of an interrupt where urgency is important. The UART interrupt is triggered when data is received over the UART, and demonstrates how interrupts can simplify your projects.

**Listing 1** shows a number of excerpts from the project. I've cut out large parts of code between the areas we need to focus on here (signified by the “=== CUT ===” inserts).

In most cases, configuring an interrupt is a simple case of setting some sort of “Interrupt Enable” bit in a relevant register. Each type of interrupt does have its own slight peculiarities though, so we'll deal with the details of the ones we're using here.

### Configuring the PCINT Interrupt

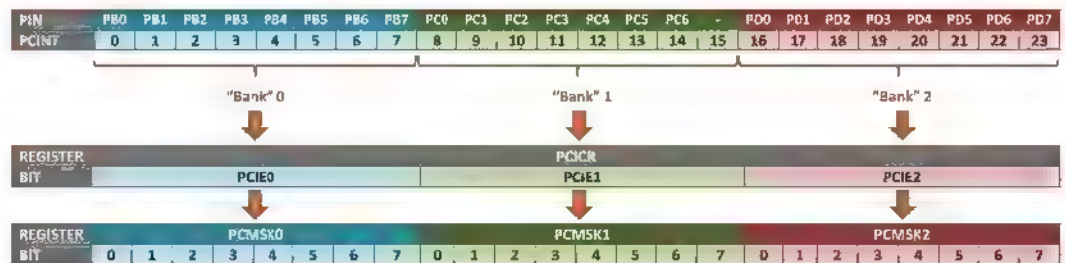
By now, you'll probably be (reluctantly) comfortable with the fact that the ATmega328P datasheet is your go-to when you need to work with new functionality on the MCU. Open it up and head over to section 13.2 — the part of the datasheet describing the registers needed for managing external interrupts. Here, you'll find details of the:

**PCICR (Pin Change Interrupt Control Register)**, which allows the enabling and disabling of interrupts when the PCINT pins toggle.

**PCIFR (Pin Change Interrupt Flag Register)**, which indicates whether an interrupt on a PCINT pin has been triggered.

**PCMSKx (Pin Change Mask Registers)**, enabling you to set which individual PCINT pins trigger an interrupt.

You may remember that the 24 PCINT pins are grouped into three banks of eight pins (although you may notice that PCINT15 doesn't exist): PCINT0-7, PCINT8-15, and PCINT16-23. **Figure 6** shows the relationship between the physical pins, the PCINT designations, the



**Figure 6: Pin-change interrupts: Pin groupings and interrupt registers.**

groupings into “banks,” and then finally how these are dealt with in the registers.

To decode all of this, let's look at what we need to do to enable the interrupt on pin **PD6**. From **Figure 6**, you'll see that **PD6** falls into “bank” 2. To enable interrupts on bank 2, we set bit 2 of the PCICR register:

```
PCICR |= (1<<PCIE2);
```

Okay, we've enabled interrupts on bank 0 — but which of the eight pins (PCINT16-23) will actually fire? For this, we use the **PCMSKx** register. There are three of these registers — one for each bank — but, of course, we only need to manage bank 2. Setting a “1” for a specific pin allows that pin to trigger an interrupt. So, the code we need is:

```
PCMSK2 = (1<<PCINT22);
```

That's all it takes to configure the pin-change interrupt. Now whenever the logic level on pin **PD6** changes, the interrupt will fire. For a button, this means that the interrupt will trigger on both the press and the release, as well as if the button bounces. We work around this in the ISR as we only want the first press of the button to trigger an interrupt in this game.

### Configuring the UART Rx Interrupt

From **Listing 1**, you'll see that configuring an interrupt to fire whenever a byte is received on the UART is really straightforward — set one bit in the *USART Control and Status Register B*, and away you go:

```
UCSR0B |= (1<<RXCIE0);
```

If you are working on a project that expects to receive a large volume of continuous data over the UART, then using an interrupt like this is probably best for receiving the first character only — and then handing it off to a routine that will receive the rest of the data. Of course, this depends on the complexity, clock speed, and processing requirements of the rest of your project. I'd suggest you experiment with both methods to see what works best for that specific application.

## Parts List

The projects are based on the build from “Beyond Arduino #1” in the March 2015 edition of *Nuts & Volts*. The following additional components are needed for the “Finger on the Buzzer” project:

D1	Green LED 20 mA
D2	Red LED 20 mA
R1,R3	330 ohm Resistors, 0.25W
SW1	Momentary Pushbutton Switch



## 12.1 Interrupt Vectors in ATmega48A and ATmega48PA

Table 12.1. Reset and Interrupt Vectors in ATmega48A and ATmega48PA

Vector No.	Program Address	Source	Interrupt Definition
1	0x000	RESET	External Pin, Power-on Reset, Brown-out Reset and Watchdog System Reset
2	0x001	INT0	External Interrupt Request 0
3	0x002	INT1	External Interrupt Request 1
4	0x003	PCINT0	Pin Change Interrupt Request 0
5	0x004	PCINT1	Pin Change Interrupt Request 1
6	0x005	PCINT2	Pin Change Interrupt Request 2
7	0x006	WDT	Watchdog Time-out Interrupt

Figure 7: Section of the datasheet's Interrupt Vector table.

For our project here — where we only expect a single character at infrequent intervals — an interrupt is the perfect solution.

### Enable Global Interrupts

The final step is to enable global interrupts. When describing global interrupts, the datasheet refers to setting the Global Interrupt Flag, or sometimes the “I” bit in the **SREG** register to a “1.” To make it simpler to read code, I prefer to use the macro *sei()* to enable global interrupts and *cli()* to disable them. I mentioned these statements earlier in the article, and you’ll see that the *sei()* statement is the last one before we launch into the main *while(1)* loop in Listing 1.

## Where did that Interrupt Go?

So, we’ve set the interrupts and they’re happily firing away when we press buttons or receive data, but where are they firing to? You’ve probably guessed from the earlier parts of the article that we need to write ISRs.

If you jump to the end of Listing 1, you’ll see the two functions named *ISR*. These are the functions that handle the interrupts, with the parameter specifying which interrupt each *ISR* handles. This may look a little cryptic — what exactly do *USART\_RX\_vect* or *PCINT2\_vect* mean, and how are we supposed to know that we need to use this syntax?

### Vectors

The datasheet refers to interrupt vectors which are basically the memory addresses the code execution jumps to when an interrupt occurs. The Atmel AVR microcontrollers use an interrupt vector table (a table of interrupt types and addresses) — refer to section 12.1 of the datasheet. Figure 7 shows an excerpt, with

vector 6 being the one we’re interested in for our pin-change interrupt. However, this table doesn’t give us the actual *PCINT2\_vect* syntax we need in our code. Atmel Studio is quite helpful with its auto-complete and does guide you towards the correct syntax. However, the definitive reference is the file *iom328p.h* — you’ll find it under your dependencies

tree in Atmel Studio’s solution explorer (Figure 8). Open this up and scroll down to a section labelled “Interrupt Vectors” — here, you’ll find the vector names that you should use in your *ISR* definitions (Figure 9).

### The ISR

Now that we’ve dealt with the vectors and naming the *ISR*, the rest is pretty straightforward. Each of the *ISRs* sets a flag to indicate to the main loop that an interrupt has occurred. Additionally:

- The *PCINT2* *ISR* disables further pin-change interrupts in order to prevent the interrupt firing on a signal bounce and on the button release. We re-enable this once we’ve finished processing the player’s button press.
- The *USART\_RX* *ISR* reads a byte from the *UART* data buffer (compare this to the code from last month where we had to check a register before reading from the *UART*).

The variables used in the *ISR* are globally defined (and using the *volatile* keyword). You’ll note that we’ve adhered to our principles and kept the *ISR* as short as possible. With the flags set, we can exit the *ISR* and leave the main loop to handle the heavy-lifting.

## While We’re Interrupt Free

One of the benefits of interrupts is that your main *while(1)* loop becomes clean and uncomplicated. While complicated code may look impressive and may give the author a sense of achievement, I am in the camp that feels simple code is better. Simple code is easier to maintain, easier to hand over to others to work on, and less likely to contain obscure bugs that are hard to track down.

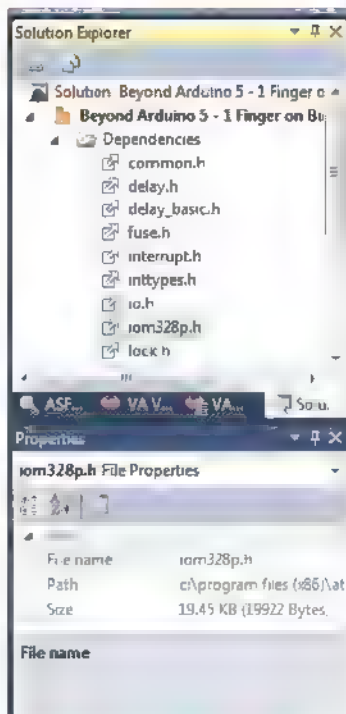


Figure 8: “iom328p.h” in the Solution Explorer.

Atmel illustrates the *volatile* keyword:  
<http://bit.ly/1zC003o>

Author's website  
[www.crash-bang.com](http://www.crash-bang.com)

The flags set in the ISR provide a consistent and easy-to-read way to handle events, and complex polling is not needed.

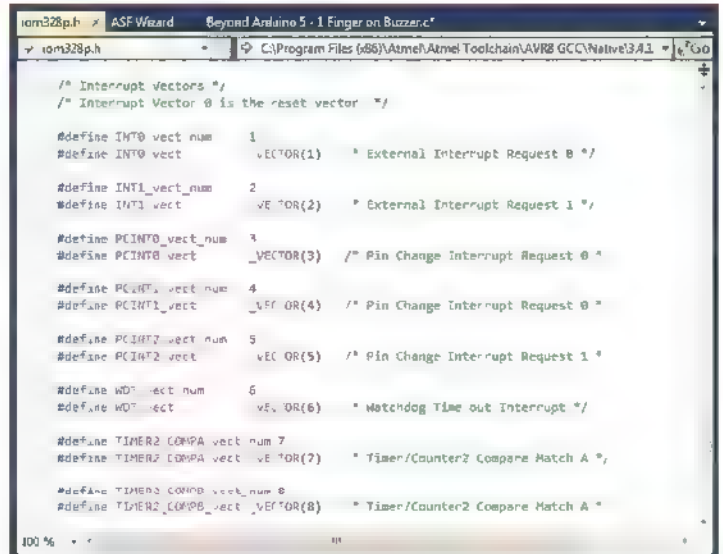
To keep the main loop simple to read, I've dedicated separate functions to handling the processing around the button press and the receipt of data over the UART.

So, go ahead. Download the code, compile it, upload it to your breadboard project, and dig into its workings. Make changes, tweak, and modify it, and, of course, please continue to share what you've achieved.

## Looking Ahead

This month's article should have given you the foundation to start experimenting with other types of interrupts. If you're looking for inspiration, why not try to "upgrade" last month's thermostat to use interrupts? The UART interrupts should be straightforward, but you'll need to spend some time in the datasheet to write one for the ADC.

Next month, we're going to spend some time working



```
/* Interrupt vectors */
/* Interrupt Vector 0 is the reset vector */

#define INT0_vect_num 1
#define INT0_vect _VECTOR(1) /* External Interrupt Request 0 */

#define INT1_vect_num 2
#define INT1_vect _VECTOR(2) /* External Interrupt Request 1 */

#define PCINT0_vect_num 3
#define PCINT0_vect _VECTOR(3) /* Pin Change Interrupt Request 0 */

#define PCINT1_vect_num 4
#define PCINT1_vect _VECTOR(4) /* Pin Change Interrupt Request 0 */

#define PCINT2_vect_num 5
#define PCINT2_vect _VECTOR(5) /* Pin Change Interrupt Request 1 */

#define WDT_vect_num 6
#define WDT_vect _VECTOR(6) /* Watchdog Time out Interrupt */

#define TIMER2_COMP_vect_num 7
#define TIMER2_COMP_vect _VECTOR(7) /* Timer/Counter2 Compare Match A */

#define TIMER2_COMPB_vect_num 8
#define TIMER2_COMPB_vect _VECTOR(8) /* Timer/Counter2 Compare Match A */
```

Figure 9: The Interrupt Vector section of "iom328p.h."

on the I<sup>2</sup>C protocol. This will allow us to start adding additional modules to our projects: real-time clocks, EEPROM modules, humidity sensors, and a host of others. Until then, learn to enjoy interruptions! **NV**



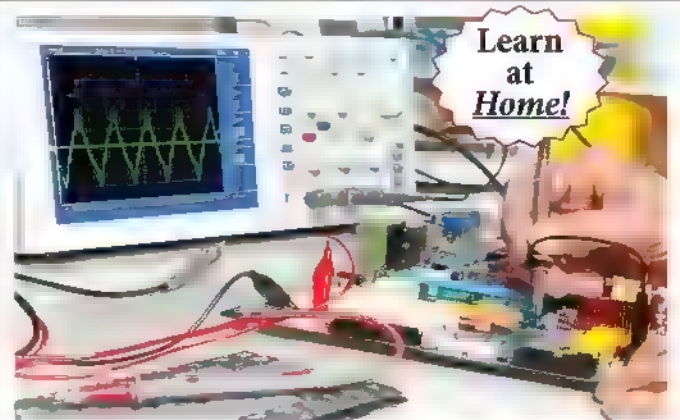
As low as...

**\$9.95**  
each!

Two Boards  
Two Layers  
Two Masks  
One Legend

**Unmasked boards ship next day!**

[www.apcircuits.com](http://www.apcircuits.com)



Learn  
at  
Home!

## Electronics Courses

Cleveland Institute of Electronics

Train at home to be a professional *electronics* or *computer* technician! Learn with hands-on labs, instructor support, online exams, videos and instructor led chat rooms.

**FREE** course catalog [www.cie-wc.edu](http://www.cie-wc.edu) or (800) 243-6446

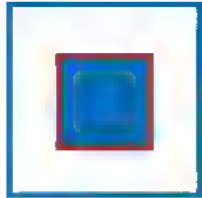
- NEW! Robotics Automation Lab
- Industrial Electronics with PLC
- Electronics with FCC
- Electronics Troubleshooting
- NEW! Computer Security Specialist
- Broadcast Engineering
- PC Troubleshooting
- Networking

Visit [www.cie-wc.edu](http://www.cie-wc.edu) and start today!

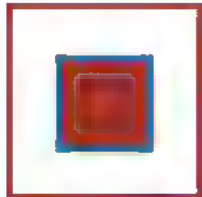
CIE 776 E 17th St. Cleveland, OH 44114 • (800) 243-6446 • Registration 70 11 0002H



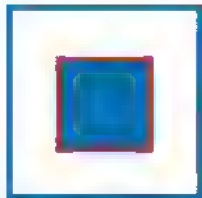




# FPGAs



# for the



# Hobbyist

Microcontrollers are very popular with electronics hobbyists, and why not? You write software that tells the hardware what to do — complete with a wide range of peripheral features and capabilities. It is not unlike programming your home computer. Hackers have done amazing things with them, pushing the performance envelope beyond what they were designed to do. However, what is even better than writing software for hardware? How about writing hardware? How about telling the hardware what to be versus what to do? Enter the exciting world of Field Programmable Gate Arrays (FPGAs)!

## Introduction

We live in a renaissance for the electronics hobbyist with new and exciting products, kits, development boards, and software hitting the “streets” daily (okay, online stores). Whether it is the Arduino, the Raspberry Pi, RTL-SDR, or the multitude of other products, there is literally something for everyone. Lately, this includes various *affordable* FPGA development boards. In the past, FPGAs were traditionally the domain of electrical engineering companies with significant resources and capital, but now these boards are opening up a new and exciting world to the hobbyist.

There are several options for those curious about FPGAs. The two largest FPGA companies and industry leaders — Xilinx and Altera — have development kits available for their various product families from \$180 and up. Digilent — probably most famous with hobbyists for their chipKIT PIC32 Arduino boards — has several options for as low as \$90. These development boards usually include additional hardware such as switches, LEDs, and VGA connectors to aid in your experimenting.

However, recently there have been several offerings

from independent developers, such as the Papilio (as low as \$40) and the Mojo (\$75). These boards have limited additional hardware; however, they are typically open source and are easily extensible. This article will focus on the latter — the Mojo V3 from Embedded Micro.

In this article, I will not cover the details of how a FPGA works in regards to LUTs, slices, etc. Nor will I cover an introduction to Verilog. Instead, this article will get you up and running with FPGAs — no prior experience required. Once you are comfortable with the basics of setting up the software and loading and running a circuit on the FPGA, I recommend you reference the *Nuts & Volts* article, “Up The Logic Food Chain” (August 2008) for a more detailed discussion on the ins and outs of FPGAs.

## You Gotta Have Mojo, Baby

The Mojo V3 board is built around the Xilinx XC6SLX9 which is part of the low cost Spartan-6 family. Sporting over 9,000 logic cells, 16 DSP slices, and 102 I/O pins (84 available for use on the Mojo V3), there is plenty of room to learn! An integral part of the board design is an Atmel

AVR ATmega32U4 used to configure the FPGA, USB communications, and to assist with analog-to-digital conversion (ADC). Additionally, there are eight LEDs and a pushbutton on the board available to the user.

The Mojo V3 arose from a Kickstarter project and boasts both open source software and hardware. Be aware, you have to supply a micro USB cable and an optional 5-12 VDC power supply if you need to draw more current than your USB port can safely provide (typically 500 mA)

## Where Do I Begin?

There is a significant software stack in order to begin using the Mojo V3. However, all of it is free of charge. The first order of business is to download ISE Design Suite 14.7 from Xilinx (if you cannot download the full DVD ISO, Xilinx will mail you a DVD free of charge). Unfortunately, the software is only for Windows and Linux, as the world of FPGAs is not yet Mac friendly. As a Mac user myself, I got around this by running Linux Mint 17.1 in Parallels Desktop 10 and it works flawlessly.

If you don't want to shell out the money for Parallels Desktop, you should have no problem running Linux Mint in VirtualBox. (I have been using VirtualBox for a Windows XP virtual machine for years.) Installation is fairly easy, but be aware that it does take a significant amount of space (20+ GB). For a detailed installation tutorial, please see Embedded Micro's *Installing ISE Tutorial*.

The next piece of software you'll need is the Mojo Loader. This software is produced by Embedded Micro and – once again – is only for Windows and Linux. As with ISE Design Suite, Embedded Micro has an excellent installation tutorial.

Optional software includes the Arduino IDE (integrated development environment) and the Mojo IDE. If you are savvy, you can load the onboard ATmega32U4 with custom software to utilize the full potential of the microcontroller for whatever you desire. I have not tried the Mojo IDE which is designed for development outside of the ISE Design Suite (although it is still required for the command line tools), as it is still very much in beta.

## Project Setup Basics

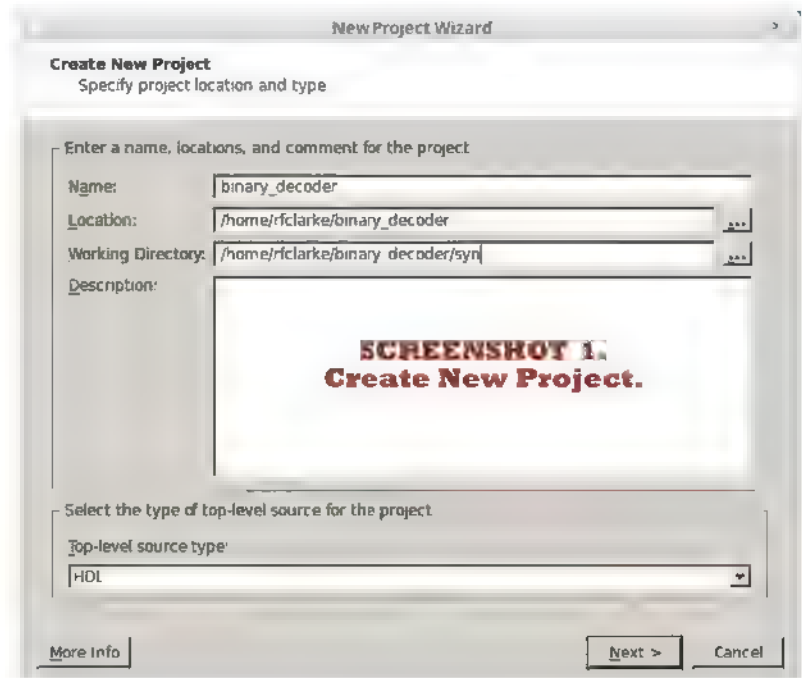
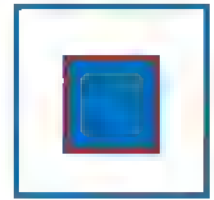
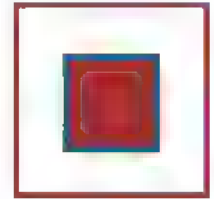
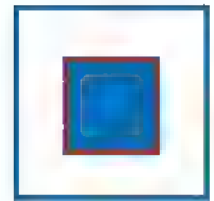
Okay, with our software stack installed, let's design something! I think an easy starting point is a 2-to-4 line single bit decoder with Enable.

This project will allow us to investigate some basic combinational logic using the Mojo V3. However, before we can start writing hardware (sounds weird, doesn't it?), let's cover the basic steps to create a project in ISE Design Suite.

With the software open, create a new project (**File->New Project...**). Name the project *binary\_decoder* and add */syn* to the end of the Working Directory (trust me on this... ISE loves to make files and this will contain the mess). Click **Next (Screenshot 1)**.

We need to specify some project settings. The Family should be set to "Spartan6" and the Device to "XC6SLX9." The Package is "TQG144" and the Speed is "-2." Now, you're probably asking me, "How do you know the speed is -2?" Great question!

If you look at the FPGA package on the Mojo V3 board, the last line of text says "2C" (**Photo 1**). Reading the *Spartan-6 Family Overview Datasheet*, this indicates a speed grade of "-2" and a commercial temperature tolerance.



## RESOURCES

Papilio  
<http://papilio.cc>

Embedded Micro  
<https://embeddedmicro.com>

Diligent FPGA Boards  
[www.digilentinc.com/Products/Catalog.cfm?NavPath=2,400&Cat=10&FPGA](http://www.digilentinc.com/Products/Catalog.cfm?NavPath=2,400&Cat=10&FPGA)

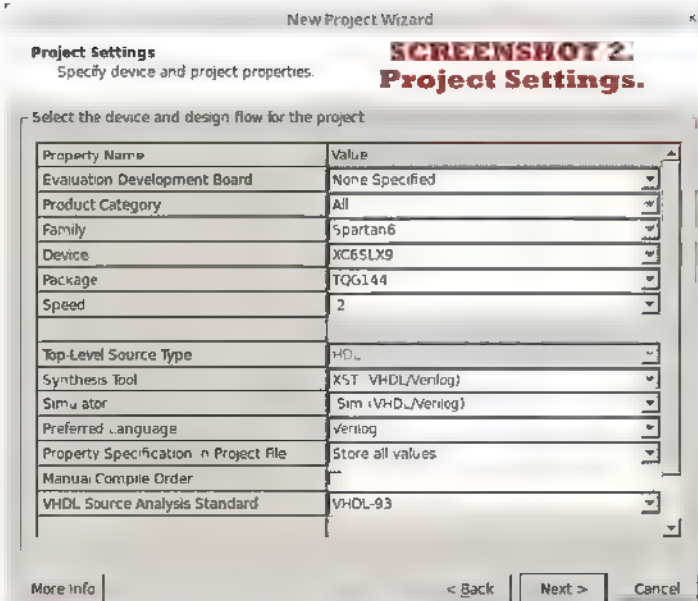
Xilinx ISE Design Suite 14.7  
[www.xilinx.com/support/download/index.html/content/xilinx/en/downloadNav/design-tools.html](http://www.xilinx.com/support/download/index.html/content/xilinx/en/downloadNav/design-tools.html)

FPGA NES  
<http://danstrother.com/fpga-nes/>

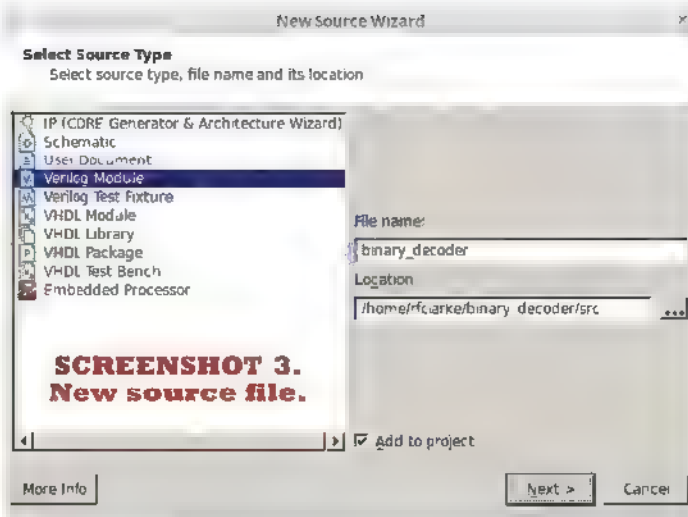
Parallels Desktop  
[www.parallels.com/products/desktop/](http://www.parallels.com/products/desktop/)

FPGA Apple II+  
[www.cs.columbia.edu/~sedwards/apple2fpga/](http://www.cs.columbia.edu/~sedwards/apple2fpga/)





**SCREENSHOT 2.**  
**Project Settings.**



**SCREENSHOT 3.**  
**New source file.**

asks to create the `src` directory, click **Yes**.

Okay, okay. I know you're chomping at the bit to start coding, but there is only one more step I promise! Select `binary_decoder` in the Implementation Hierarchy, look below at the Processes tree, and right-click on Generate Programming File. Select Process Properties, check Create Binary Configuration File, and click **OK**. Done! Let's write some hardware!

## Technologic

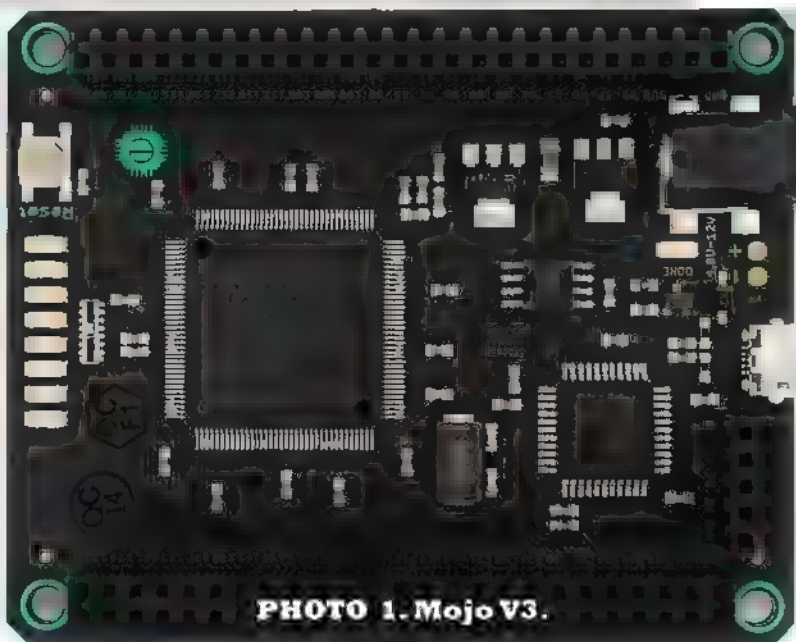
Okay, don't get mad, but ... I may have lied to you back there. Before we can start coding, we need to first look at the truth table for a 2-to-4 line single bit decoder (**Figure 1**). It's pretty simple, really. The two-bit binary input, `A`, corresponds to a specific bit in `D`, which is four bits wide. For example, an input of `A = 10` and `EN = 1` creates an output of `D = 0100`. In other words, `A = 2` corresponds to `D2 = 1`. The logic gate diagram for the truth table is shown in **Figure 2**. This is pretty easy to write in Verilog, so let's do it! Copy the following Verilog code into the source file we just created — `binary_decoder.v`:

```
// gate-level 2-to-4 line decoder
// with enable

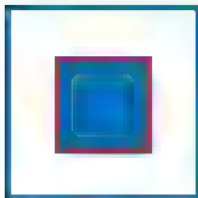
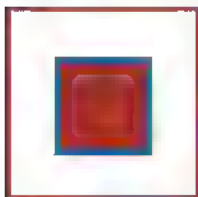
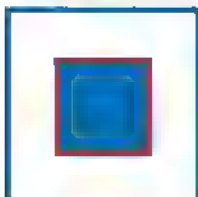
module binary_decoder
(
    input wire en,
    input wire [1:0] a,
    output wire [3:0] d
);
```

EN	A1	A0	D3	D2	D1	D0
0	X	X	0	0	0	0
1	0	0	0	0	0	1
1	0	1	0	0	1	0
1	1	0	0	1	0	0
1	1	1	1	0	0	0

**FIGURE 1. A 2-to-4 line binary decoder with Enable truth table.**



**PHOTO 1. Mojo V3.**



Lastly, verify that "Verilog" is selected as the Preferred Language. Click **Next** (**Screenshot 2**). Click **Finish** on the next window, titled Project Summary.

As stated, we'll be designing a 2-to-4 line single bit decoder with Enable. Let's create the Verilog source for the decoder. Create a new source (**Project>New Source**), select Verilog Module, and name the file `binary_decoder`. Add `/src` to the end of the Location and click **Next** (**Screenshot 3**). At the next window — titled Define Module — just go ahead and click **Next**; we'll define our I/O ports later. In the Summary window, click **Finish**. When it

```

assign d[0] = en & ~a[0] & ~a[1];
assign d[1] = en & a[0] & ~a[1];
assign d[2] = en & ~a[0] & a[1];
assign d[3] = en & a[0] & a[1];
endmodule

```

While this is not a tutorial on Verilog itself, let's break down our code real quick. The first portion of the module defines the inputs and output. We have a single bit input labeled *en* and a two-bit input labeled *a*. The four-bit output is labeled *d*. The last portion simply defines the output as the set of four three-input AND gates, with the appropriate inputs inverted per the truth table. Easy!

Okay, let's actually define our I/O pins. Create a new source, but instead of a Verilog Module, define it as an Implementation Constraints File. Name it *binary\_decoder* and copy the following code into the source file:

```

CONFIG VCCAUX = 3.3;

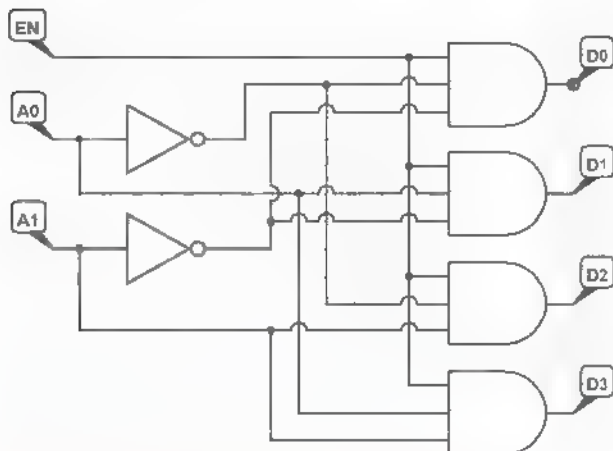
NET "en" LOC = P139 | IOSTANDARD =
LVCMOS33;

NET "a[0]" LOC = P137 | IOSTANDARD =
LVCMOS33;
NET "a[1]" LOC = P138 | IOSTANDARD =
LVCMOS33;

// Mojo V3 LED 2-5
NET "d[0]" LOC = P134 | IOSTANDARD =
LVCMOS33;
NET "d[1]" LOC = P133 | IOSTANDARD =
LVCMOS33;
NET "d[2]" LOC = P132 | IOSTANDARD =
LVCMOS33;
NET "d[3]" LOC = P131 | IOSTANDARD =
LVCMOS33;

```

This file defines which pins are associated



**FIGURE 2. The 2-to-4 line binary decoder with Enable circuit diagram.**

with the I/O ports in our modules. The first line is related to how the Mojo V3 is designed. The VCCAUX pins of each of the I/O banks are tied to 3.3V. Long story short, this means we are restricted to 3.3V logic. The other lines should be fairly self-explanatory.

That's it! How about we go for a test drive?

## Simulation

Before we generate the programming file, let's take a look at ISim in order to verify that our Verilog code works. To do this, we need to create a Test Bench. Create a new source file, define it as a Verilog Test Fixture, and name it *binary\_decoder\_test*. Copy the following source code into the newly created file:

```

`timescale 1 ns / 10 ps

module binary_decoder_test;
// test inputs and outputs
reg en;
reg [1:0] a;
wire [3:0] d;

// unit under test
binary_decoder uut(.en(en), .a(a),
.d(d));

// run tests for all iterations of
// en and a, 250 ns per test
initial begin
en = 1'b0;
a = 2'b00;
# 250;

en = 1'b0;
a = 2'b01;
# 250;

en = 1'b0;
a = 2'b10;
# 250;

en = 1'b0;
a = 2'b11;
# 250;

en = 1'b1;
a = 2'b00;
# 250;

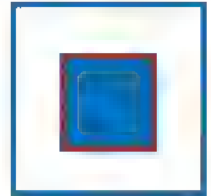
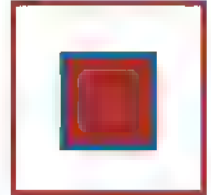
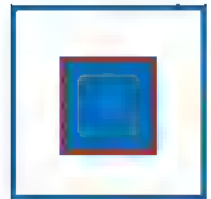
en = 1'b1;
a = 2'b01;
# 250;

en = 1'b1;
a = 2'b10;
# 250;

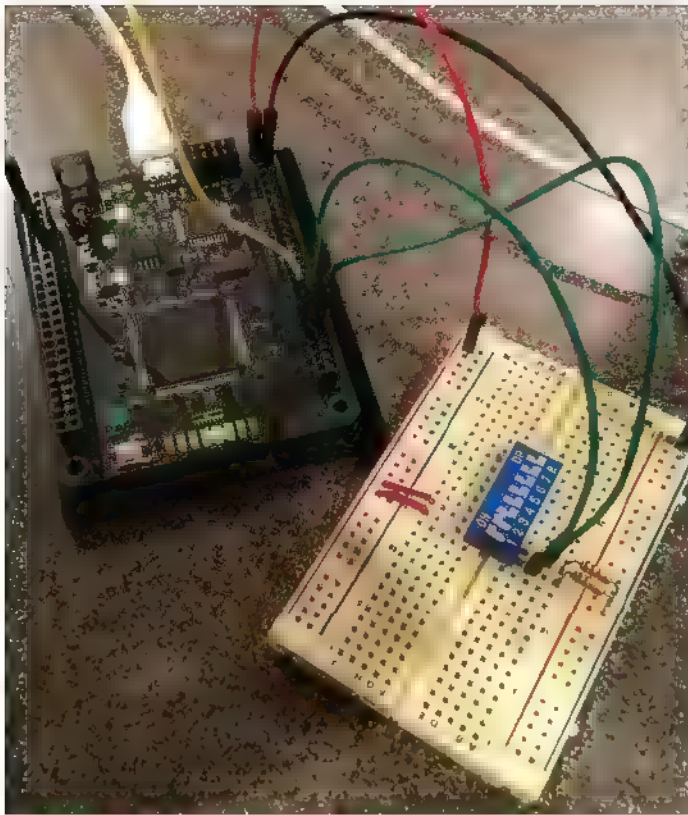
en = 1'b1;
a = 2'b11;
# 250;

$stop;
end
endmodule

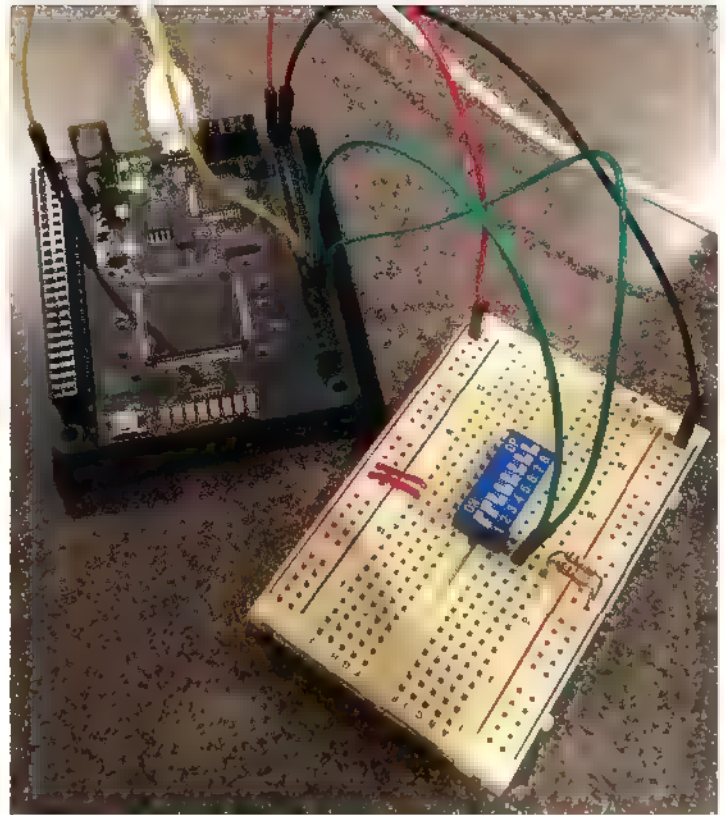
```







**PHOTO 2. Circuit test: EN = 1; A = 11; and D = 1000.**



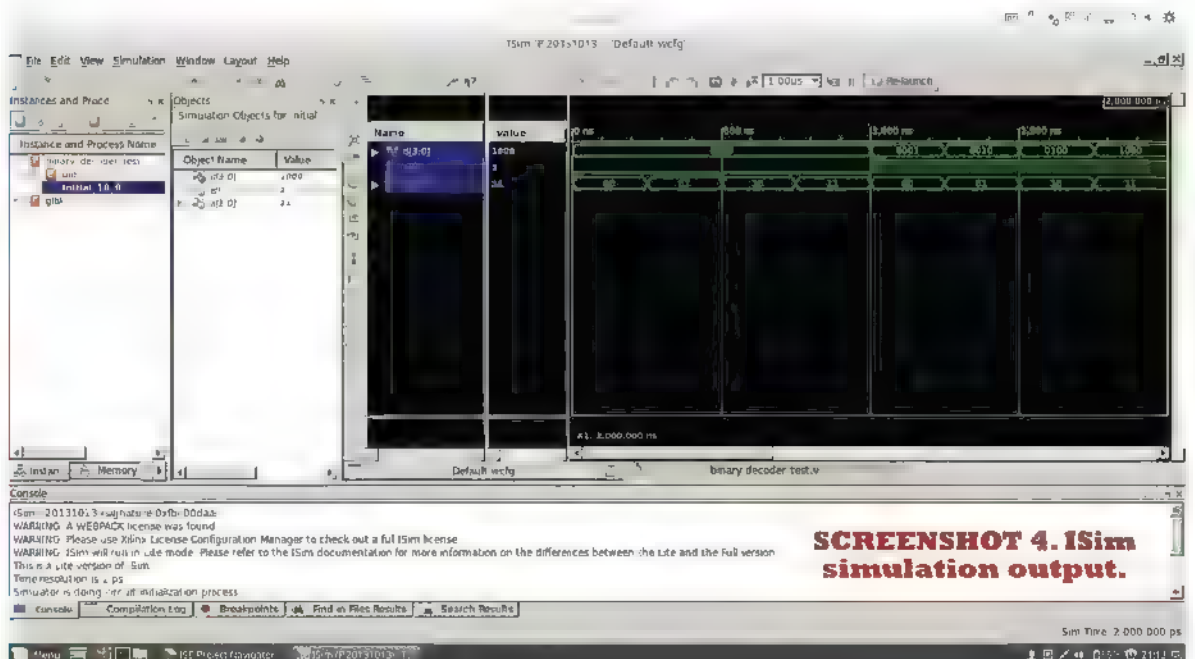
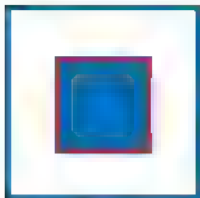
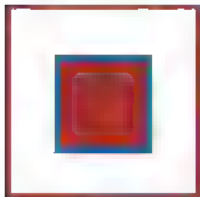
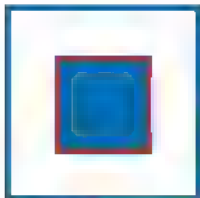
**PHOTO 3. Circuit test: EN = 0; A = 11; and D = 0000.**

The quick and dirty version of what this does is as follows: We define a set of registers and a wire that correspond to the inputs and output of our device. The Test Bench is then written to test each input iteration of *en* and *a* at a 250 ns interval.

I'm sure your next question is, "How do we

see the results?" Another great question! With Simulation view selected in ISE Design Suite, select *binary\_decoder\_test* in the Behavioral Hierarchy and double-click on Simulate Behavioral Model in the Processes tree below. The ISim software should open automatically.

Select the **Simulation->Run All** menu



**SCREENSHOT 4. ISim simulation output.**

Ryan Clarke graduated from Worcester Polytechnic Institute (WPI) in 2003 with a B.S. in Electrical Engineering, concentrating in Signals and Communication. While at WPI, he was inducted into Eta Kappa Nu (HKN), the IEEE electrical engineering honor society. Ryan also holds an Amateur Extra FCC amateur radio license. Upon graduation, he commissioned into the US Navy as an Ensign and is currently a Naval Aviator flying F/A-18F Super Hornets.

option and then reselect the *Default.wcfg* window. Select the **View->Zoom->To Full View** menu item and you should now be able to view the results of the Test Bench (**Screenshot 4**). As you can see, our implementation of the 2-to-4 line decoder with Enable works as advertised.

## Synthesis

Okay, time for the fun part. Let's load the Mojo V3 and test the hardware! Close ISim and go back to the Implementation view. Select *binary\_decoder* and in the Processes tree, double-click on Generate Programming File. It takes a surprising amount of time to generate the binary file for the FPGA, so be patient. Once it is complete, you should find the *binary\_decoder.bit* file in the *syn* sub-directory of your root project directory. Following Embedded Micro's *Creating a Project* tutorial, connect your Mojo V3 and program the FPGA.

To test the circuit, I wired up an eight-pin DIP switch with 10K pull-down resistors to easily toggle the inputs. Position "1" corresponds to  $A_0$  and position "2" corresponds to  $A_1$ . As you can see in **Photo 2**, with an input of  $A = 11$ , the output is  $D = 1000$  as expected! **Photo 3** is the same value of  $A$  with EN tied to ground, therefore disabling the output.

## Now What?

To learn more, I recommend the book *FPGA Prototyping By Verilog Examples* by Dr.

Pong P. Chu. This is the book I am currently working through myself, and it is very easy to follow. In fact, the example circuit we wrote in this article is one of the first suggested exercise experiments by the author.

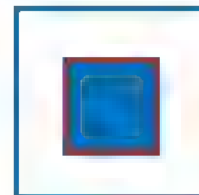
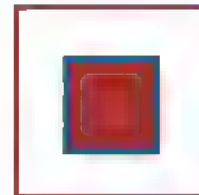
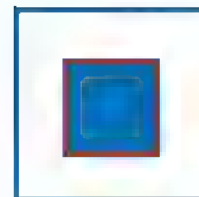
While the book is written for an older Digilent Spartan-3A development board, it is fairly easy to port them over to the Mojo V3 or any other FPGA board you might be using. You may have to supply some extra hardware, but you probably have most of it lying around your junk box. Of note, if VHDL is your thing, Dr. Chu has an identical book written for that particular language.


So, you may be asking yourself, "What else can I do besides combinational logic?" The answer is ... virtually anything in the digital world! Want to generate VGA or NTSC signals? No problem (with the help of some digital-to-analog conversion, of course). How about a custom microcontroller core? Yup. An entire recreation of the Nintendo Entertainment System? It's been done! An IBM PC or Apple II? Absolutely!

The learning curve is steep — especially if you are accustomed to procedural programming and microcontrollers. However, once you wrap your mind around how the FPGA operates, its parallel structure, and have an understanding of digital logic, the possibilities are truly endless!

■ ■ ■

*If you don't want to manually type in the code presented here, it is available at the article link*





# PBP3

**PICBASIC PRO™ Compiler 3.0**  
BASIC Compiler for Microchip PIC® microcontroller

~~FREE~~

Student - ~~\$49.95~~

Silver Edition - \$119.95

Gold Edition - \$269.95

**A world class BASIC Compiler for rapid development of Microchip PIC® microcontroller based projects.**

- Lightning Fast
- Generates Optimized, Machine Ready Code
- Easy enough for the hobbyist, strong enough for a pro.
- This is a full-blown development tool that produces code in the same manner as a C compiler (without the pain of C).
- It is very easy to learn and understand.

Download a FREE Trial Now!

www.PBP3.com

PICBASIC and PICBASIC PRO are trademarks of Microchip Technology Inc. in the USA and other countries. PIC is a registered trademark of Microchip Technology Inc. in the USA and other countries.



# An Ultra Modern Shortwave Radio

A simple circuit, a USB TV tuner, your computer, and some powerful software combine to make an amazing software defined communications receiver.

By George R. Steber

This unusual radio can be used to receive conventional amplitude modulation (AM) and frequency modulated (FM) stations, as well as more specialized modes such as narrow band FM (NFM), single-side band (SSB), continuous wave (CW), and other signals. Its performance – while good – is slightly below the best shortwave listening (SWL) receivers or ham radios. However, we will show you how to improve its performance substantially with tunable filters

As it stands, it does a creditable job in many situations and has features that many other radios do not – like digital signal processing, spectrum analysis, and a waterfall display. As an added bonus, you will be able to receive frequencies from 24-1766 MHz in case you tire of the classic shortwave frequencies.

In today's jargon, this receiver would be called a software-defined radio (SDR) – a concept made popular by the military. Typically, in an SDR receiver, an analog-to-digital conversion (ADC) is made on the RF signal, and the rest of the functions of a classic analog receiver are performed on the digital signals in software. These functions include tuning, filtering, and demodulation. Using software to replace hardware allows more versatility, provides more functions, and reduces hardware complexity.

For this design, we will use a simple RF converter and a small inexpensive commercial digital TV module to

provide the up-front hardware. The rest of the functions will be done on the computer.

So, if you want to explore the world of shortwave radio and learn more about this technology, read on and definitely consider building your own software-defined receiver.

## Software-Defined SW Receiver

**Figure 1** is a block diagram of our shortwave SDR receiver. Starting from the antenna, there is an RF converter/mixer block. The purpose of this block is to perform frequency up-conversion of the shortwave signals. This is necessary because the next block requires RF signals above 24 MHz. Therefore, a simple up-conversion of the signals is performed using a mixer. Also in this block are some analog filters to prevent strong AM and FM signals from overloading the mixer and causing intermodulation distortion.

The next block is a digital TV tuner – an amazing device that does most of the work. It is connected to the RF up-converter with a short coaxial cable to its antenna terminal. In this device, analog signals from the up-converter are tuned, changed to digital signals, and conveyed via USB to the last block.

The final block is your home PC which performs all the necessary digital signal processing. It should be

It was the simplest of times — it was the most complex of times. Indeed, much like those words, this shortwave radio project is probably the simplest, most complex one you have ever seen. It is simple because the main hardware components are low cost, readily available, and easy to assemble. It is complex because you need to download and install a special USB driver and other software on your PC — and get it all working together. Once the components are assembled and the software is installed, however, you will have a very versatile shortwave radio that covers the 1-30 MHz range, and provides many enjoyable hours of shortwave listening and experimentation. Since the software is free and the components are cheap, you should be able to build this receiver for around \$25.

moderately fast, have some USB ports, and a sound card. The input data to the computer is entirely in digital form, coming from the TV tuner via USB. After applying digital algorithms for filtering and demodulation in the PC, the processed digital data is presented to the sound card — all in the digital domain. Finally, the sound card is used to convert the digital data to audio to drive speakers or headphones. Before we continue with details of the design, let's take a closer look at the little tuner device that makes this project possible.

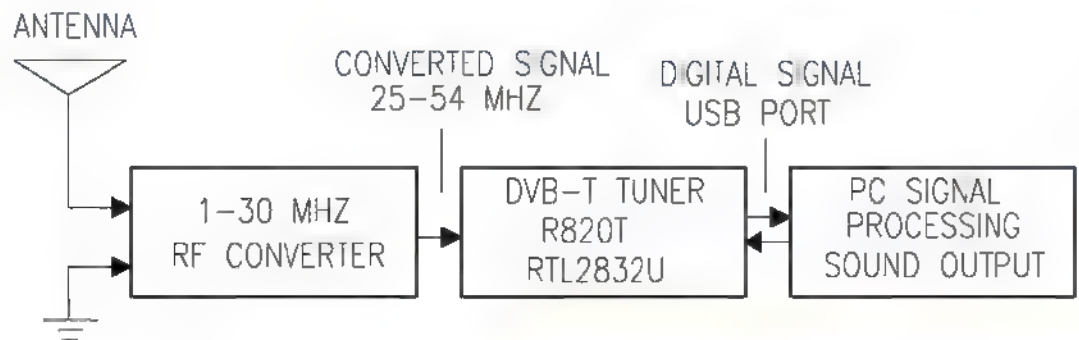


FIGURE 1. Block diagram of SDR SW receiver.

## Mini DVB-T USB Tuner

The digital terrestrial TV age is in full bloom for millions of viewers around the world — especially in Europe. Analog terrestrial TV has been replaced by the new Digital Video Broadcasting Terrestrial (DVB-T) standard in many places. Viewers use mobile USB TV tuners with their computers for reception of HDTV broadcasts. Mobile tuners have compact dimensions — about the size of a memory stick. They plug into and are powered by the USB 2.0 port of your computer.

Figure 2 is a photo of a DVB-T tuner package purchased on eBay for about \$11 (including shipping). As you can see, a remote control and an antenna are included — ideal for watching TV on your laptop.

We will not be using the DVB-T tuner for TV viewing in this project. Besides, it would not work in the US since the DVB-T standard is not implemented here. Instead, it will be repurposed to act as a wide-band tunable SDR receiver. Hence, we will not need the other items in the TV package.



FIGURE 2. Low cost DVB-T package.





**FIGURE 3.** RTL tuner (with R820T and Realtek RTL2832U inside) and BNC adapter cable.

Figure 3 is a photo of the DVB-T stick we are using and a short interface cable that was bought separately. Most DVB-T sticks have two main chips inside: a digitally controlled tuner and an ADC that samples the baseband signal and outputs the samples to a host computer through a USB port.

A very specific DVB-T stick is used in this project — one that has a Rafael Micro R820T tuner IC and a Realtek RTL2832U inside. These so-called RTL sticks are quite common and available from many sources. Beware,

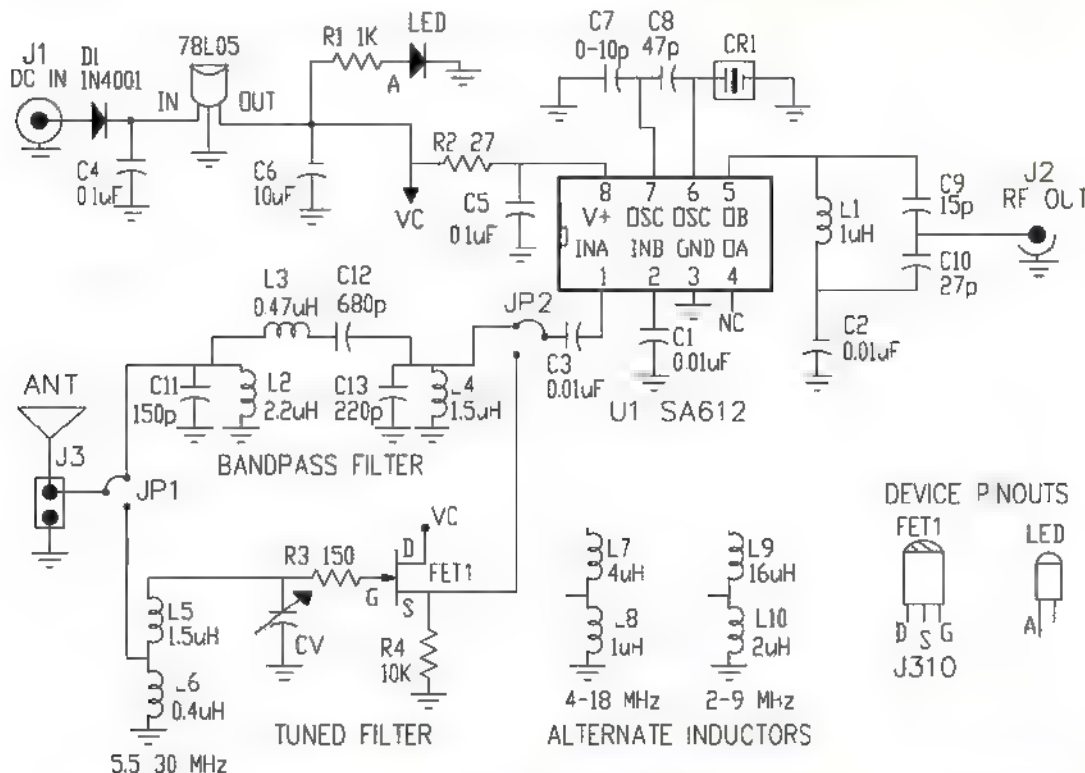
though, as some sticks have an E4000 tuner or some other kind of tuner IC inside. Those *will not* work in this project. Make sure your stick has an R820T inside.

The R820T tuner is crucial because it has the lowest tuning range — tunable down to 24 MHz. Other tuners do not have that low-end range. As we will see later on, this capability makes our shortwave up-converter easy to build.

What makes the RTL stick so valuable is its inherent ability to demodulate FM signals and transfer the amplitude and phase information as raw in-phase and quadrature phase (I/Q) samples to the computer via USB. Annti Palosaari — a Finnish engineering student and Linux developer — discovered this special radio mode. Amazingly, this mode enables the tuner to output a stream of eight-bit I/Q samples at rates up to two million samples per second.

Once this discovery was grasped, enthusiasm grew for the development of a cheap SDR. The group from Open Source Mobile Communication (Osmocom) — particularly Steve Markgraf — developed a basic set of drivers and utilities to communicate with the RTL dongle. After that, other software developers began writing code to use these drivers and provide user interfaces. One of those programs, SDR# (pronounced SDR Sharp) is probably the most popular one. We'll discuss it in more detail later on.

## RF Converter and Assembly Details



**FIGURE 4.** Detailed schematic — shortwave radio RF up-converter.

The front end of our shortwave receiver is a frequency up-converter. An up-converter is a circuit that adds a constant frequency to the received frequency — the one received at the antenna. An RF mixer and local oscillator can be used to make a simple up-converter. If we use a local oscillator frequency of 24 MHz, then the output of the mixer will contain the received signal plus 24 MHz. Both the local oscillator and the mixer functions can be handled by a general-purpose SA612 device.

If you search the Internet, you will see some designs that use

down-converters with 125 MHz local oscillators. These circuits are overly complicated and harder to build. They are also more prone to spurious signals and more costly. Our little up-converter costs quite a bit less and performs better in most instances.

The design of the frequency converter is straightforward. However, there is one slight wrinkle as two different front-end filters are presented: broadband and tunable. This gives diverse users the option to utilize the SDR for different applications. The broadband filter is low cost and works well over the entire 1.5–30 MHz range, but has more noise due to inter-modulation from strong in-band stations. This option works well for strong AM SW stations from around the world without the bother of peaking a filter.

On the other hand, the tunable filter has a narrower band and requires manual signal peaking, but has less noise and can dig out the weak stations. This option is better for receiving amateur radio and other low power stations.

**Figure 4** is the circuit for the 24 MHz up-converter. A parts list for the project is shown in **Figure 5**. Most parts for the receiver are readily available from sources like Mouser, Digi-Key, and Jameco. The SA612 is the only chip in the circuit, and is used for mixing and local oscillator functions. The up-converter frequency is determined by crystal CR1. It needs to be 24 MHz or slightly higher. The exact frequency is not critical, as an offset frequency will be used in the final software to compensate. So, any frequency in the range of 24 MHz–24.6 MHz will work fine. It is important that the capacitors C7 and C8 be good quality — like Class 1 NPO ceramic or even silvered mica — to minimize temperature drift. C7 is probably not needed. Leave it out if the circuit oscillates without it.

The circuit is powered from eight to 12 volts DC and is regulated via the 78L05. Use a linear voltage supply as opposed to a switching type to avoid noise. Batteries work well too, as the current draw is small.

As mentioned earlier, there are two RF input filter choices. Choosing the upper jumpers on JP1 and JP2 connects the bandpass filter (BP). You don't actually need to build both filters — only build the one you want. The BP filter is designed to remove strong stations from the AM broadcast band and FM band that would otherwise overload the mixer.

The other choice is the tuned filter. It requires hand wound high Q coils and a variable capacitor to peak the signal. (See the Parts List for details.) For this filter, only specific bands are covered. So, with this option, if you want to cover the whole HF range, you will need to

## Parts List

Qty	Label-Value	Designation(s)	Description
3	0.01 $\mu$ F	C1, C2, C3	Mono capacitor
2	0.1 $\mu$ F	C4, C5	Mono capacitor
1	10 $\mu$ F	C6	Electrolytic
1	0-10 pF	C7	Class 1 NPO ceramic- see text
1	47 pF	C8	Class 1 NPO ceramic
1	15 pF	C9	Ceramic capacitor
1	27 pF	C10	Ceramic capacitor
1	150 pF	C11	Ceramic capacitor
1	680 pF	C12	Ceramic capacitor
1	220 pF	C13	Ceramic capacitor
1	1K	R1	1/4W 5% resistor
1	27	R2	1/4W 5% resistor
1	150	R3	1/4W 5% resistor
1	10K	R4	1/4W 5% resistor
1	1 $\mu$ H	L1	Fastran RF inductor, Mouser
1	2.2 $\mu$ H	L2	Fastran RF inductor, Mouser
1	0.47 $\mu$ H	L3	Fastran RF inductor, Mouser
1	1.5 $\mu$ H	L4	Fastran RF inductor, Mouser
1	1.5 $\mu$ H	L5	15T, T50-6 core
1	0.4 $\mu$ H	L6	9T, T50-6 core
1	4 $\mu$ H	L7	28T, T68-6 core
1	1 $\mu$ H	L8	13T, T50-6 core
1	16 $\mu$ H	L9	56T, T68-6 core
1	2 $\mu$ H	L10	20T, T50-6 core
1	78L05	78L05	5V regulator, Jameco
1	J310	FET1	FET
1	LED	Green	LED
1	15-400 pF	CV	Variable capacitor
1	J1	DC	Jack 3.5 mm
1	J2	RF OUT	BNC type
1	J3	ANT	Two terminal
1	U1	SA612	RF mixer/osc, Mouser
1	D1	1N4001	Diode
1	CR1	Crystal	24 MHz - see text

**Figure 5. Parts List for SDR SW receiver.**

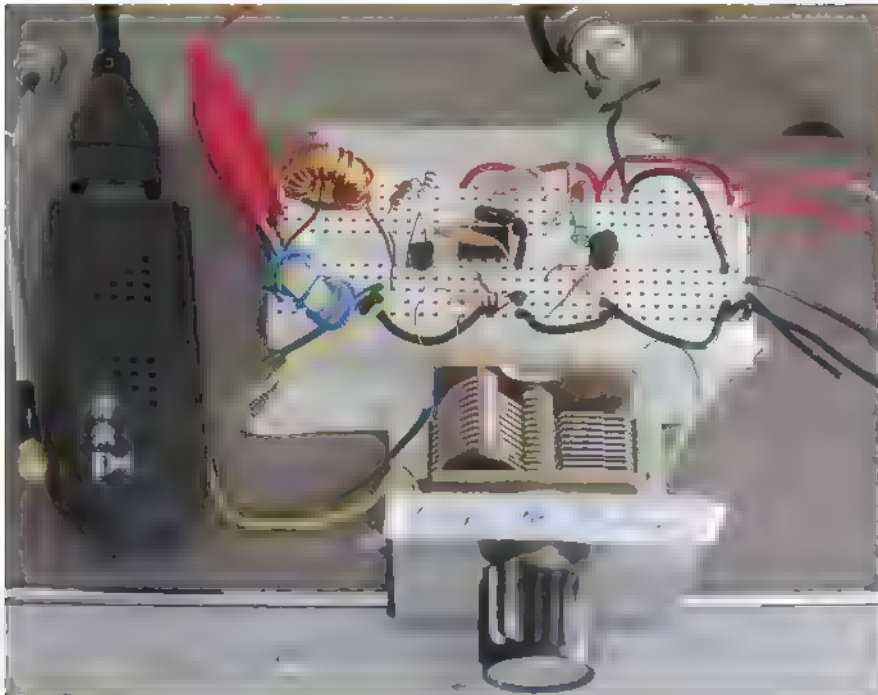
provide switching between coils. The improved performance may be worth it.

The variable capacitor CV needs to have a wide range — typically 15 pF to 400 pF. These capacitors are getting more expensive and harder to find, but can still be found on the Internet for around \$15.

The output “up-converted” signal is provided to J2 (BNC) through a tuned output stage designed to match the 75 ohm input of the RTL antenna. **Figure 6** is a breadboard for the tuned filter receiver. In general, this open layout is not recommended for RF circuits. For best performance, the mixer circuit should be built on a PCB (printed circuit board) and housed in a small metal box. Although a PCB design is not available, it should be easy to make — even for beginners.

A short coaxial cable is connected from J2 to the RTL stick's antenna terminal. The RTL usually has an MCX type connector as its antenna terminal. Because an MCX plug is very small and difficult to solder, it is suggested that you buy a ready-made adapter cable — an MCX plug with a short RG316 cable to BNC. **Figure 3** shows such an adapter cable. To reduce computer noise, a USB





**FIGURE 6. Breadboard of tunable filter shortwave radio.**

extension cable may be used to move the RTL farther away from the computer.

## Software Installation and Settings

First of all, do not use any of the software that came with your RTL device. It was designed for a different application. For our radio, the RTL stick requires a special USB driver and a graphical radio interface. You will need to install Zadig for bulk interface drivers and then SDR# for the radio interface. Once Zadig is run, you will have a new USB driver named “Bulk-In, Interface (Interface 0).” That is what the RTL device uses.

After you install SDR#, you will be able to select this driver to use with your radio. Also note that – depending on your PC operating system – you may need Microsoft’s Net 3.5 for the installation to work.

The software installation discussed earlier is too detailed to present here. The following websites can take you through the procedure. Perhaps the best site that covers the entire installation is [www.rtl-sdr.com/rtl-sdr-quick-start-guide](http://www.rtl-sdr.com/rtl-sdr-quick-start-guide). Other sites to check are <http://rtlsdr.org/softwarewindows> and [http://inst.eecs.berkeley.edu/~ee123/fa12/rtl\\_sdr.html](http://inst.eecs.berkeley.edu/~ee123/fa12/rtl_sdr.html). The installation may seem a bit daunting when you are just starting out, but remember, once you get it working it will be well worth the effort.

A screenshot of SDR# tuned to WWV is shown in **Figure 7**. It’s interesting to see atmospheric fading in the

waterfall plot. Running SDR# is like having a radio lab at your disposal. More details on its operation can be found at <http://sdrsharp.com>.

Now, let’s take a look at some of the screen buttons. In the upper left corner is the Play button – which obviously starts the program. What is not obvious is that some settings cannot be changed while it is running. So, if you cannot set something, make sure the program is stopped. The first thing you should do is select the USB driver you installed by pressing the down arrow located next to the Play button. Choose RTL-SDR/USB from the available options.

Next, click Configure. Choose a sample rate of 1.024 MSPS and Quadrature sampling. Leave the other boxes blank. Notice that there is a slider to select the RF gain. Set it to about 14.4 dB. It can be changed later if it’s too low. There is another box at the bottom that allows correction for the crystal frequency inside the RTL. We will discuss that later.

Now, look below the Play button.

There are numerous buttons for selecting NFM, AM, and so forth. In this screen area is a box labeled Shift. Click this box and enter: 24,000,000. Notice the ‘minus’ sign. This number corresponds to the crystal frequency in the up-converter and will provide the first rough correction – it enables SDR# to read out the RF SW frequency directly.

Now, move down to the Audio section. The Samplerate and Input boxes should be grayed out. This is because data is being sent via USB not audio. In the Output box, select the sound card that you are using. It will produce the sound you will hear.

Let all of the other settings in SDR# be at their default for now. You will have lots of fun experimenting with them later on. Connect a long wire antenna and ground counterpoise. Make them as long as possible – 25 feet or more. With any luck, you should now have a functioning SW radio.

Three automatic gain controls (AGC) are available. The ones in the Configure box do not work well. The AGC on the main screen works well on strong stations. For weak stations, leaving AGC off and manually adjusting the RF gain may work best.

Other SDR# settings are usually a matter of preference, but here are some guidelines. Generally, a moderate value of Zoom should be used unless you are calibrating the radio. This makes it easier to click on a peak to select a station. Use a small number for FFT – typically 4096 to get good computer performance. Use a bandwidth that is appropriate for the signal being monitored: CW 800 Hz; SSB 2.8 kHz; and AM 10 kHz. A

filter order of 40 works well and will have little effect on program speed.

Tuning in an SSB station can be a challenge as there is no carrier peak. Tune to the left or right of the signal depending if USB or LSB is used, and adjust the frequency dial slowly until voice is intelligible.

You may want to calibrate the frequency dial more accurately. This is a two-step procedure as there are two crystal oscillators used here: one in the RTL and one in the RF converter. To calibrate the RTL, uncheck Shift and connect a short wire of three or four feet directly to an RTL antenna terminal. This should allow you to pick up a NOAA weather station in your area — usually around 162.400 MHz. Look up exact frequencies on the Internet. Note that NOAA uses NFM. Click Configure and adjust the Frequency Correction (PPM) until the dial reading corresponds to the NOAA frequency.

To calibrate your HF SDR receiver, check Shift and tune to station WWV using USB — not AM. WWV at 10 MHz is useful for calibrating. Use a high value of FFT resolution — around 131,073 — so you can see WWV peak clearly in the spectrum. Adjust the Shift value until the dial reads the peak frequency accurately to within 50 Hz. It will drift periodically, but that is the nature of uncompensated crystal oscillators.

As noted above, if you bypass the HF converter and use an antenna directly connected to the RTL, you can take advantage of the very large frequency range of this device — up to 1,766 MHz. Thus, you may be able to directly pick up FM stations (and other stations like NOAA) and two-meter ham repeaters if they are close enough.

## Final Comments

Hopefully, by now you should be having fun with your SDR SW receiver. If you are having difficulties, go to <http://sdrsharp.com> or the Yahoo SDR group and they may be able to help.

Shortwave radio is more exciting now than ever before. Many new AM broadcasters from China, Cuba, Europe, and other places will keep you informed and entertained. Religious broadcasters also abound. Unfortunately, many of the interesting stations only appear at night due to propagation conditions. Fortunately, the powerful ones come booming in even in the daytime.

Listening to ham radio operators using SSB and Morse

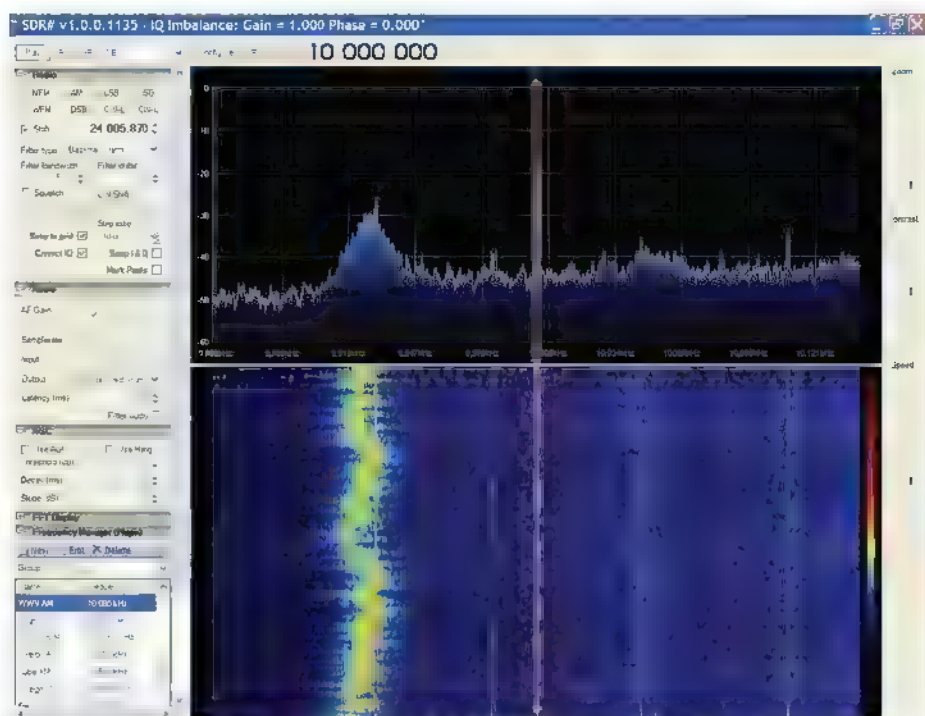


FIGURE 7. SDR# opening screen — radio tuned to 10 MHz WWV.

code is also fun — especially during contests. Other stations are using SSB too, such as Maritime WX on 8.763 MHz and Aviation WX on 10.051 MHz. These generally use synthetic voices. If you are lucky — as I was — you may hear one of the mysterious 'numbers' stations using USB around 13.199 MHz. However, they are seldom in the same spot twice. Also interesting are utility stations such as WLO, which transmits on many frequencies and provides high seas communication and weather information.

One of the exciting things you can do with this radio is receive data transmissions such as WX FAX, SITOR, RTTY, BPSK, WSPR, and EasyPal SSTV. These modes can easily be decoded with available software, but to discuss this further would require another magazine article.

If you like playing around with this radio and find the subject fascinating, you may want to consider becoming a radio amateur. Hams are involved with building and studying receivers, transmitters, antennas, satellites, EME, microwaves, and experimenting with new radio modes such as WSPR, BPSK, Packet Radio, and more. If you are considering joining the fraternity of radio amateurs, the ARRL website may be the place to start. Be sure to check out the new column here in *Nuts & Volts*, as well.

In the meantime, have fun with your new SW radio.





# Reviving a Hi-Fi Classic:

## *the Harman/Kardon A-401 Stereophonic Control Amplifier*

By J.W. Koebe



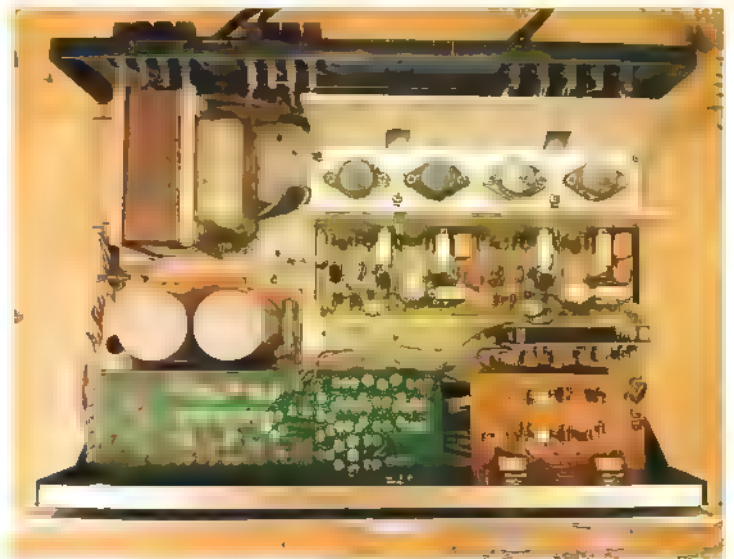
Vintage stereo equipment just has a special sound and presence unlike anything that's made today, and these pieces of hi-fi history are always worth preserving and restoring. I recently got to work on a classic from 1978: the Harman/Kardon A-401 stereophonic control amplifier.

I found this one nearly dead. It powered on, but that was about it. Both channels crackled and sputtered wildly, and barely any music made it from the inputs to the speakers – much longer and it would surely have suffered a catastrophic failure that destroyed it permanently.

This particular little amplifier offers 20W of power at less than 0.5% THD and is a great example of elegant understated styling from the late '70s. It's not the most powerful amp out there by any means, but it fits nicely in a room and has that classic hi-fi sound you just can't get with anything else.

### Inside the Amplifier

The A-401 stereophonic control amplifier was designed in the USA and manufactured in Taiwan, and was designed with both sound and serviceability in mind which made it a real pleasure to work on. It features a pre-



The dirty insides of this 38 year old amp.

A quasi-complimentary output stage is a type of push-pull output stage made from two identical NPN transistors connected between the supply rails, in contrast to a fully complimentary output stage which uses a mirrored pair of a PNP and NPN transistors. Quasi-complimentary designs were popular in the early days of transistor amplifiers, when NPN power transistors were easier to manufacture than PNP power transistors, but continued to be found in lower powered and entry level amplifiers for many years. They're almost never used in modern construction as NPN and PNP power devices are equally available.

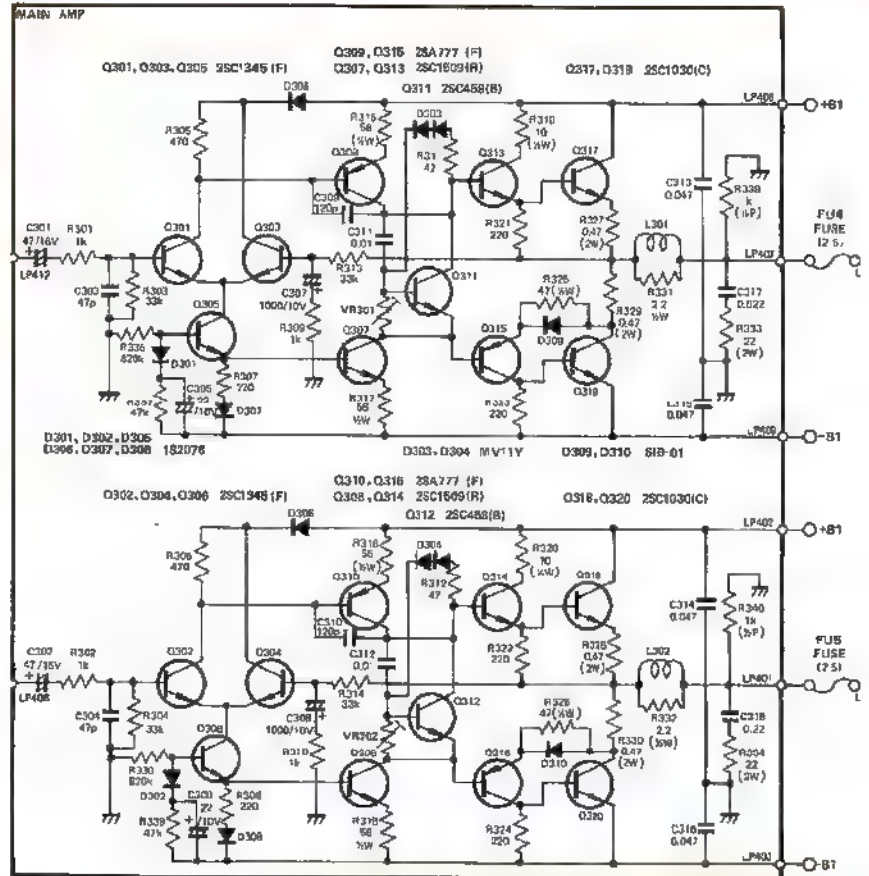
amplifier stage designed for a magnetic phono cartridge, tone controls, and quasi-complimentary output stage with a differential pair input to help keep distortion down.

This one was fairly dirty inside and the 2SC1030 output transistors were badly corroded, but it survived in this condition for quite a while. This was a good example of Harman/Kardon not messing with a good thing, too: The phono amplifier board and final board show up in the model 430 receiver, and likely many others from around the same time.

Except for the phono board, everything can be re-worked without even removing it from the chassis which made this one really easy. The first task was to shotgun replace all of the electrolytic capacitors on every circuit board. These capacitors start to dry out or leak with age – without fail – and can cause all kinds of havoc in the circuit. Without known good capacitors in place, there's no point in doing any other work. I replaced all the small capacitors with Nichicon Fine Gold audio-grade capacitors

– all de-rated for longevity – and Panasonic snap-in capacitors for the main power supply.

It's tough to find exact replacements for some of the components used in these vintage amplifiers – especially filter capacitors. It's easy to find something that's



Partial schematic of the final amplifier board.



This phono amplifier PCB shows up in several other H/K amplifiers and receivers.



The tone and phono boards showing new Nichicon Fine Gold audio-grade electrolytic capacitors.





Showing the final power amplifier board with freshly replaced capacitors.



Components have come a long way in the last few decades! Left, the original snap-in capacitor; right, a drop-in replacement rated at three times the working voltage while still taking slightly less space.



New capacitors installed on the power supply board.

As with all electronics projects, take proper safety precautions while servicing a vintage stereo system. Don't do any work with the line cord connected — even if the unit is off — as power line leakage through the heating element of your iron could destroy transistors. Pay attention to unsafe vintage wiring schemes that may put mains voltage on exposed metal parts of the chassis, and keep in mind that some parts can get hot or might have exposed high voltages when operating. Be careful!

electrically compatible, but it's pretty common for the physical layout to be just slightly wrong. For example, it's nearly impossible to find screw-terminal capacitors with 10 mm lead spacing, and 12.5 mm lead spacing just won't cut it.

Miraculously, 6,800 µF 100V capacitors from Panasonic were perfect drop-in replacements for the original Elna capacitors. They're physically very similarly sized, but the new ones are rated for 100V — about triple the original 35V rating. With such a significant de-rating, they should last for decades.

The power supply in this amplifier is a bit simpler than you'd find in a bigger amp. While a larger one might use an active voltage regulator power supply, this one has a simple transformer, bridge rectifier, filter capacitors, and zener diodes for voltage regulation. It's simple but effective.

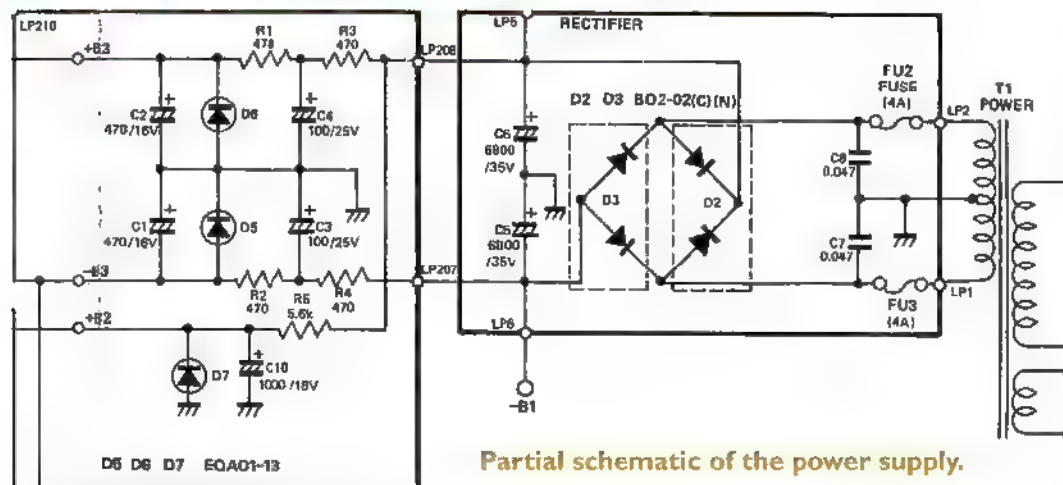
## First Power-Up

After replacing all the electrolytic capacitors, it was time to power it up for the first time and see how it was feeling.

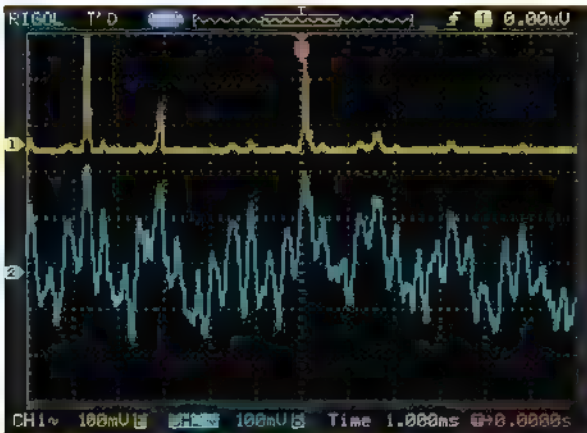
No smoke! One side was working great, but the other

still had some issues. In addition to still sputtering badly when starting up, it was terribly distorted to the point of being almost unrecognizable. Time to drag out the oscilloscope and see what's going on! I followed the schematic to see where the signal was distorted. The tape out jack was my first step to see the output from the pre-amp stage.

Since I was using the tuner input — which



Partial schematic of the power supply.



Oscilloscope trace showing the bad channel. Trace (1) showing the output of the defective side — only responding to high peaks and only in the positive direction, while trace (2) shows the other channel's correct output.

doesn't go through the phono amplifier — there's only the switch between the tape output and the tuner input. As expected, everything was fine on the amp's front end. This meant the problem had to be in the final amplifier.

In a push-pull amplifier design, if the transistors aren't conducting evenly you'll hear the distortion audibly. I put the scope on the speaker terminals to see what the output signal was doing while playing some music, and it was showing a clearly one-sided trace, and only the highest peaks at that.

This helped me narrow it down to the negative side of the push-pull circuit.

From the input to the final amplifier, I traced through the circuit using the oscilloscope until I found the point the signal became distorted. It turns out that Q8 — a 2SC1509 transistor for that side — was defective. It was conducting, but just barely. I pulled it and replaced it with a new old stock part, and that cleared the distortion right up. The sputtering and popping in that channel remained, though. Time for more digging!

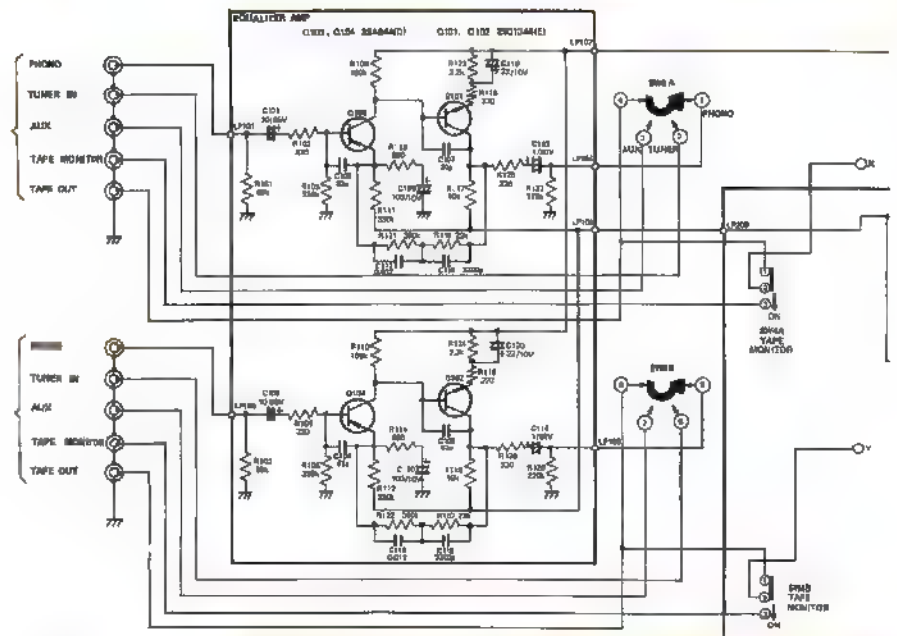
## The Plot Thickens

While poking around with the oscilloscope to figure out where the noise was coming from, another problem

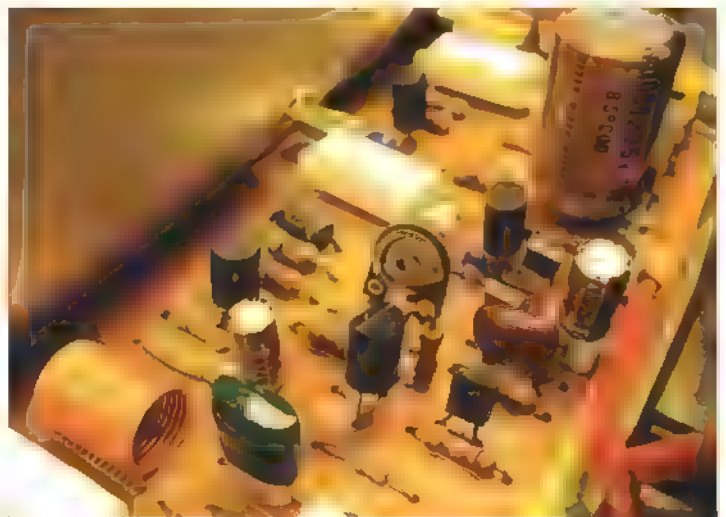
Q8, the defective 2SC1509 transistor, center.



Ready for first power-up.



Partial schematic of the inputs and phono pre-amp.





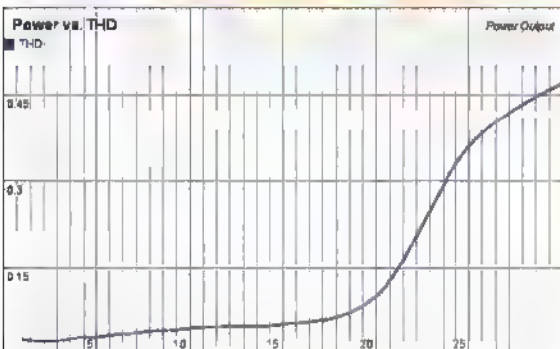


Close-up of the unmodified final amplifier board.

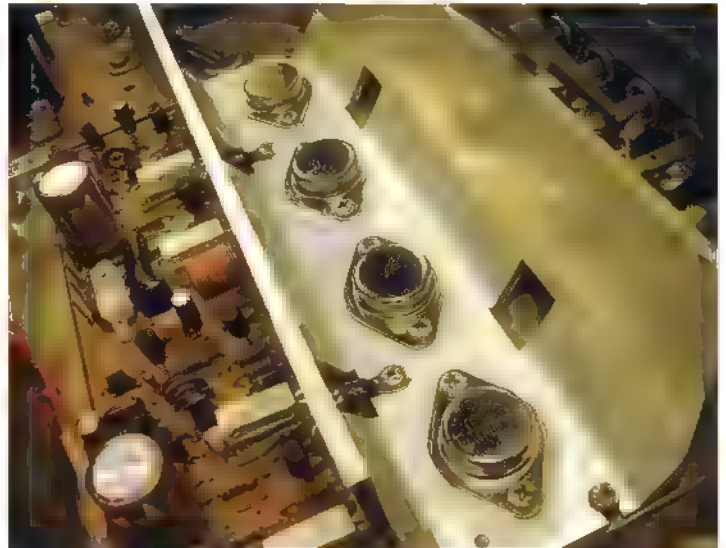


All four replaced small-signal transistors.

Bias-adjusting variable resistor.



Power output vs. THD at 1 kHz into 8 ohms.



New ON Semi MJ15003G output transistors.

unexpectedly developed: The fuses in the transformer secondary started blowing. That's very bad. It means something has developed a major short circuit and is drawing far too much current through the transformer secondary. Thinking it was a fluke or I'd touched something wrong, I replaced the fuses and fired it up again with all my probes removed. It lasted two or three seconds before popping again. Not good at all.

If the excessive current was drawn through the resistors in the low voltage section of the power supply, they'd have smoked immediately, but those resistors were all fine — this meant the failure was in the output transistors which are directly connected right after the transformer and bridge rectifier. I pulled them for inspection.

The transistors themselves all tested good — somehow — but the interesting part was their insulators. The collector of the 2SC1030 in this is tied to the metal case, mounted to a grounded heatsink. An insulator failure would short the collector (and supply rail) to ground, which is no good. As luck would have it, one of the mica insulators looked slightly cracked and had a carbonized trace on it. That'd be the problem, although it picked a weird time to show up.

I replaced all four mica insulators, applied new thermal grease, and for good measure I replaced the original 2SC1030 with all new ON Semiconductor MJ15003G transistors. These are drop-in substitutes, but like the capacitors are very de-rated in this application and should be reliable for a long time to come.

## Back On Track

After I popped in new fuses again, the next power-up was stable and I was back where I'd started: one channel

sounding great, but the other still showing a bit of sputtering and popping that just wasn't going away. It was worst at power-on and settled down a little over the next few minutes, but never disappeared entirely. Transients like that are a bit tough to track down using a scope since they're so quick. I traced forward from the input, stopping each time I saw spikes on the scope corresponding with pops in the speakers, and replaced them each one at a time. With each replacement, the noise cleared up a bit until it was completely gone. Three more transistors were bad: Q306, a 2SC1345; Q312, a 2SC458; and Q310, a 2SA777.

After the last replacement, I fired it back up and it sounded crisp and clean just like it should! Problem solved. Time to finish up!

## Finishing Up

With the electrics sorted, it was time to go through final checks. This amplifier has a very simple design, so there's no power supply reference or DC offset adjustments to make; just the idle current adjustment across R329/R330 for  $25\text{ mV} \pm 1\text{ mV}$ .

Then, I ran some power curves. The combination of top-of-the-line audio grade electrolytic capacitors, new output transistors which are being run well below their maximum ratings, and good old fashion hi-fi design meant this one really shined. This little amplifier delivered below 0.1% THD through its rated power (20W) but continued to meet its power spec all the way to maximum output at 30W, while still staying below 0.5% THD.

This is pretty impressive — 50% more available power, and 1/5th the rated distortion through its factory power output. Vintage hi-fi amplifiers are often rated conservatively from the factory, but that's a significant improvement.

With that, this H/K A-401 amplifier was ready for use as a daily driver at the heart of a nice little vintage stereo system. At only 30W/channel, it's not going to be setting any stereo competition records, but that's plenty of power for filling a room at a comfortable listening volume. It's got a great minimalist design, too, which makes it very attractive and definitely worth having fixed up.

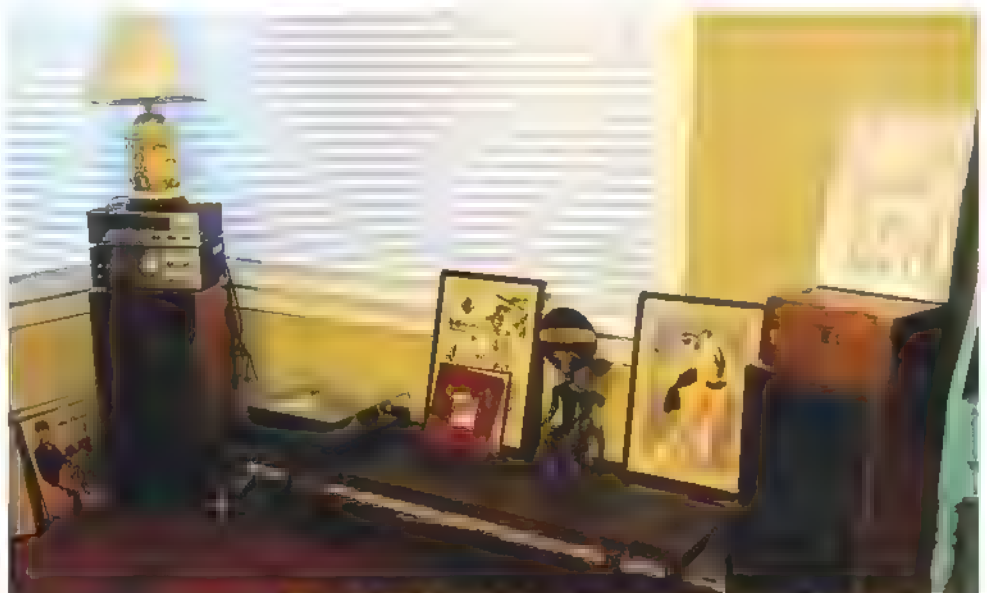
This was a pretty fun project which took some good troubleshooting. There was a



Out with the old, in with the new!  
Lots of replaced parts in this project.

defective driver transistor causing bad distortion, a failed insulator causing a dead short from the power supply, and three other intermittent small-signal transistors which were introducing noise into the signal path.

Vintage hi-fi repair is a fantastic hobby, and this project only took me about eight hours from start to finish. Replacing the electrolytic capacitors took about two hours, and the rest of the time was spent on slow and careful diagnostics through each stage. At the end, there's a beautiful functional silver front stereo amplifier to bring your music to life. Most vintage stereo repairs involve some detective work, but it's well worth it. **NV**



The A-401 back in its natural environment, powering a great little vintage system.



# Power Sources:

## THE GOOD, THE BAD AND THE UGLY

What You Don't Know about Your Power Source may be the Problem

By Eric Bogatin

### Typical Power Sources

With few exceptions, every electronics project needs a power source. It's what makes the electrons go round and round. However, not all power sources are created equal.

Too much voltage noise can screw up analog measurements. Not being able to deliver enough current when you need it means your project fails. A transient voltage drop may cause something else in your build to fail – the hardest sort of problem to debug. Delivering too much current when there is a malfunction usually means smoke is in your future.

If you know the requirements of your application and – more importantly – the characteristics of your power source, you can avoid these problems and have confidence. If there is a problem, at least it's not with your power source.

In this article, we look at four common sources of powering a project:

- Wall Warts
- Batteries
- USB Power
- Arduino Digital Pins

Even among the same type of source, there can be very big differences in their properties and performance. Let's look at the most important features we should care

about for any power source, how to measure them, and what to expect.

The poster child we'll use to explore the properties of a power source is a 12V wall wart, shown in **Figure 1**. We'll use it to illustrate the tests we'll run, and analyze the features to look for. Then, we'll apply these tests to the



**FIGURE 1.** A 12V wall wart used in these early investigations.

other types of sources and look at what to expect. You will never look at a power source the same again.

## Not All Wall Warts are the Same

Almost every low cost consumer product today uses an external wall wart as its primary power source. The output is usually a DC voltage from 5V to 30V. If you get rid of the original product, don't get rid of the wall wart. Re-use it for your next project.

I have a box of extra wall warts I've been collecting, and use them over and over again. **Figure 2** shows some of that collection. Look on the outside of any wall wart and you will see a rating for the DC output voltage and the current rating. You can see some of these values in the figure. Don't take these numbers too seriously. They are more of a guideline. Before you use the wall wart — or any power source — you'll want to characterize it using the tests described here.

I grabbed one of these wall warts that had 12V output/500 mA stamped on its side. We'll also look at what this really means and why this doesn't tell the whole story.

## Voltage Properties

To test the properties of a power source, we need to be able to measure the voltage at its output not only when it is unloaded, but also when a resistive load is attached which would draw comparable current as with the intended application.

What we want to look for is the DC voltage levels and the AC noise — usually in the form of 60 Hz or 120 Hz ripple. The best way of doing this is using two channels of a scope. If you only have one channel available, use it to measure the AC ripple and use a digital multimeter (DMM) to measure the DC voltage.

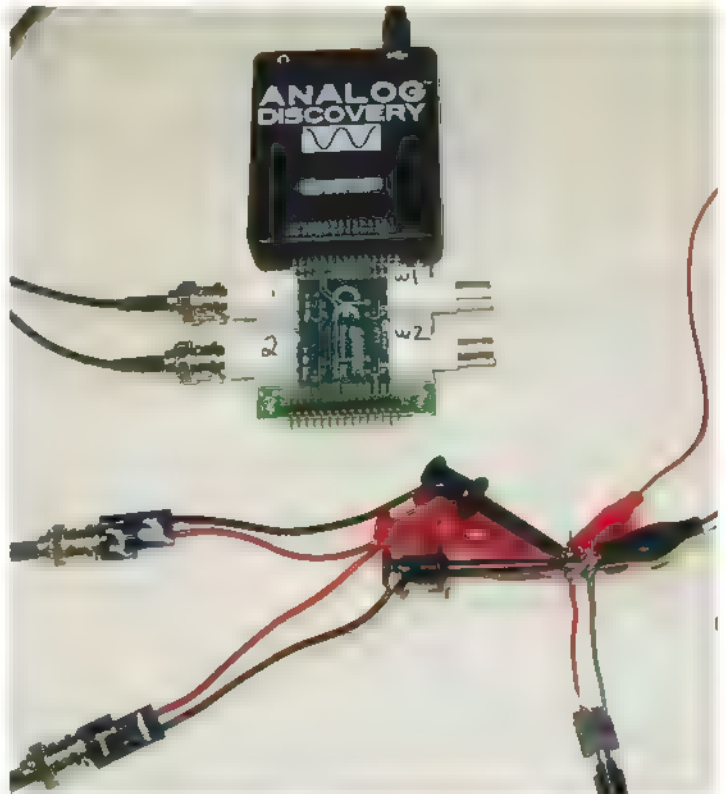
The AC coupled mode of the input to a scope is the perfect setting to use to look at 60 Hz or 120 Hz noise on the power supply voltage.

My personal favorite scope for general lab applications is the Digilent Analog Discovery scope. It has two channels — each with 14-bit vertical resolution — 100 Msamples per second, and an excellent user interface. I can add a math function which can calculate the average voltage — even the peak-to-peak voltage. On top of that, you can do an FFT (Fast Fourier Transform) of the measured voltage signal to look at the spectrum of the entire waveform.

The system I used to perform the voltage measurements in this article is shown in **Figure 3**. The scope is set up to measure the unloaded voltage from a wall wart, using a simple power plug to connect to the wall wart. Channel 1 was set for DC, 1 megohm input, and channel 2 was set for AC coupled input.



**FIGURE 2.** A selection of some of the wall warts I've collected and reused for my projects.

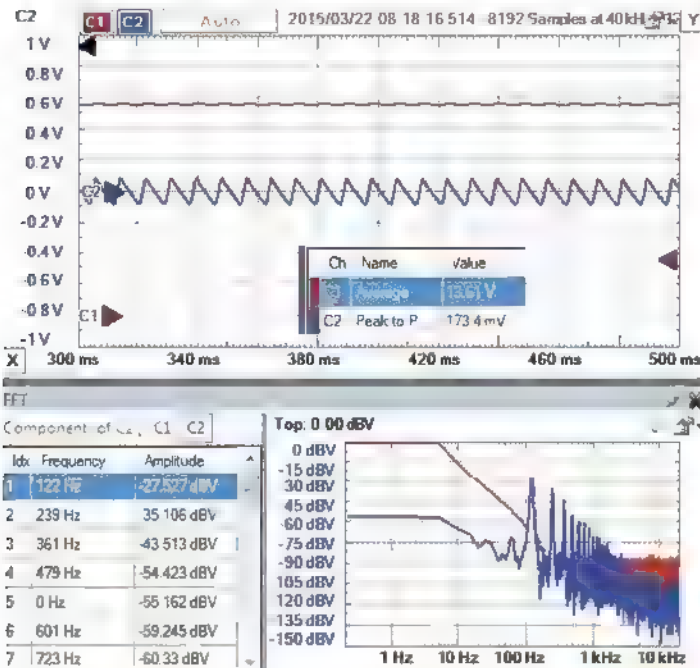


**FIGURE 3.** Analog Discovery scope with two channels connected to measure the voltage on the power rail.

## Use a Light Bulb as a Load Resistor

For future reference, if you need a 13 ohm high power resistor, a 75 watt incandescent light bulb will do. A 100 watt incandescent light bulb has a resistance of about 10 ohms. These are the resistances when cold. They increase as they get warmer. Of course, due to the internal electronics of a compact fluorescent light bulb (CFL), you can't use one as a resistive load.





**FIGURE 4.** Screenshot from the Digilent Analog Discovery scope customized to show the DC voltage, the AC voltage, the average voltage, peak-to-peak voltage, and the FFT of the AC ripple signal.

The measured DC and AC ripple from this unloaded wall wart is shown in **Figure 4**. The scope's user interface allows me to customize the information displayed to quickly get me the answer I'm looking for.

While the rated voltage is 12V, we see the measured average voltage is actually 13.61V. There is also clearly some AC ripple. In fact, the scope measures 0.173V of peak-to-peak ripple. This is a noise of about 1.3%. The DC and peak-to-peak values are read right off the math functions of the scope interface.

## How to Read an FFT Spectrum

In addition to just the voltage over time measurement, I set up the scope for an FFT analysis of the AC ripple signal. We see the waveform in the time domain is triangular, with a repeat frequency of 120 Hz. This is from a full wave rectified 60 Hz source, so it repeats every 120 Hz rather than every 60 Hz.

When we take a triangle wave and look at its spectrum, we see all the harmonics — multiples of the first fundamental frequency — with amplitudes that drop off with frequency. The spectrum shows all the peaks plotted on a log-log scale. The first harmonic is 122 Hz, the next is 239 Hz, and then 361 Hz. They are not exactly 120 Hz and its multiples because of the time window I used. With a time base of 200 msec full scale, the resolution of the spectrum is  $1/0.2 \text{ sec} = 5 \text{ Hz}$ . To within 5 Hz, they are 120 Hz, 240 Hz, and 360 Hz.

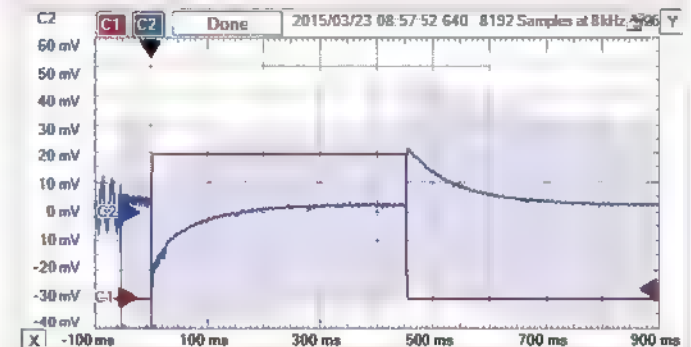
## Avoid this Cross Talk Problem when You Use the ADC Pins

When any Arduino digital pins switch, they load the internal 5V rail on the die. This 5V supply is also used as the reference voltage for the ADC (analog-to-digital converter) pins. **Figure A1** is an example of the measured voltage on pin 12, set as output HIGH, which is internally connected to the on-die 5V rail, while pin 13 is switched on.

We see the voltage noise on the on-die power rail drops as much as 20 mV when an output pin switches on. This results in noise on the reference voltage for the ADC and an artifact on the measured input voltage. This noise on the internal voltage rail will give false analog voltage readings to any of the ADC channels.

When I do precision voltage measurements, to get around this cross talk problem I use the 3.3V supply voltage on the Arduino board as the Vref for the analog pins. This gives a much cleaner and more stable analog voltage reading, but it also limits the highest voltage I can measure to be 3.3V.

In the sketch, you need to include the statement in the `void setup()` function, `"analogReference(EXTERNAL);"`. This will connect the Vref of the ADC to the external Vref pin, which I connect to the 3.3V rail of the Redboard Arduino. I've measured this as a very clean signal.



**Figure A1.** The measured voltage on pin 12 when pin 13 switches. The square signal is the voltage on pin 13 when it switched high. The signal that dips down then up is the voltage noise on pin 12, pegged HIGH. This is a direct measure of the voltage on the internal +5V rail which is also used as the Arduino's reference voltage for the ADC.

The highest frequency measured is related to the data sample rate of the scope. The Nyquist Theorem says that the highest sine wave frequency component that can be measured is half the data rate if the signal has a random noise component.

In this example, the sample rate was 40 kHz (displayed in the upper bar along the top of the screenshot), suggesting the highest frequency sine wave component we can measure is 20 kHz. This is a perfect frequency range to evaluate the ripple noise on the power supply.

The vertical scale is the amplitude of each sine wave component. I set the scale to be in dB. To convert the dB value into a voltage amplitude, we take the dB value, divide by 20, and put this to the power of 10.

The peak value of -27.5 dB for the 122 Hz component is really a voltage amplitude of  $10^{(-27.5/20)}$

= 45 mV. However, we measured a peak-to-peak value of 173 mV in the time domain. Taking half of this as the amplitude would be 87 mV – almost twice the 45 mV we get from the spectrum.

This difference is because in the time domain, we measured the total peak-to-peak value of the triangle wave; while in the frequency domain, we are looking at the sine wave frequency component amplitudes.

The first harmonic amplitude of this triangle wave is 45 mV, but there are a bunch of other amplitudes in the spectrum – each dropping off with higher frequency.

These two displays of the same information – the time domain and the frequency domain – give different pieces of information.

When evaluating power supply noise, the peak-to-peak and the amplitude at the first harmonic components are both important metrics.

## Loading a Power Supply: How Not to Blow Things Up

The ripple noise of 1.3% doesn't sound like a lot, but this is with nothing plugged into the power supply – not how it will be used. We really want to know the voltage and ripple when it is supplying current through a typical load. The challenge is applying a typical load resistance and not having it melt.

Adding a resistor across the power supply with a value to draw the current that might be typically used in our application will load the supply and test it in a more realistic situation. How much resistance should be used? This depends on the current draw in the application.

If we know the power supply voltage and the current load, we can estimate the resistor to use to draw the same current using Ohm's Law. Its three forms are:

$$V = IR \quad I = \frac{V}{R} \quad R = \frac{V}{I}$$

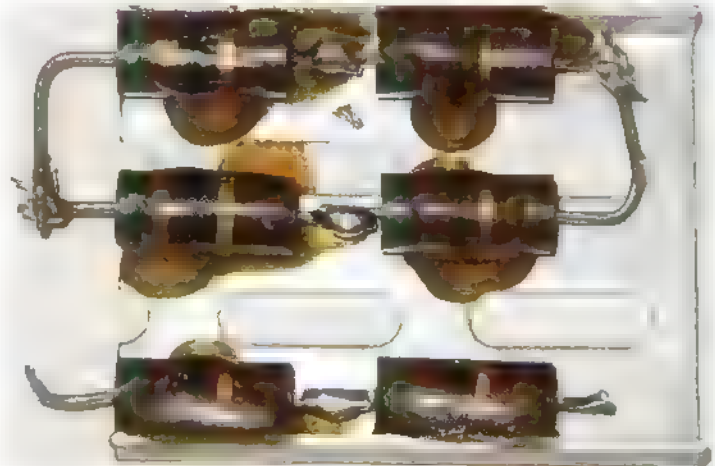
If the application calls for 12V at 500 mA, the equivalent resistive load is about  $R = 12 \text{ V} / 0.5 \text{ A} = 24 \text{ ohms}$ .

Before you immediately connect a 24 ohm resistor across the power supply to see what the voltage noise is, be sure to estimate the power consumption expected and use a resistor with a body size that can handle this power dissipation.

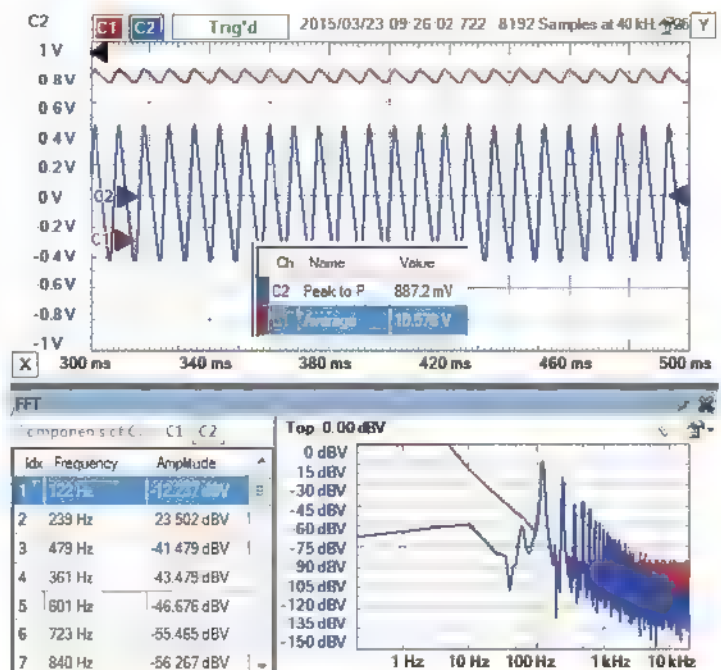
If we connect a 24 ohm resistor across this 12V supply, the power consumption would be  $12^2 / 24 = 6 \text{ watts}$ . This is way more than can be dissipated by a typical resistor.

To handle this power, I mounted six one watt 50 ohm resistors on an aluminum heat spreader as shown in **Figure 5**.

Depending on how they are connected together, I can get resistances of 10 ohms to 300 ohms. Two resistors



**FIGURE 5.** An array of one watt resistors on a heat spreader capable of dissipating 10 watts.



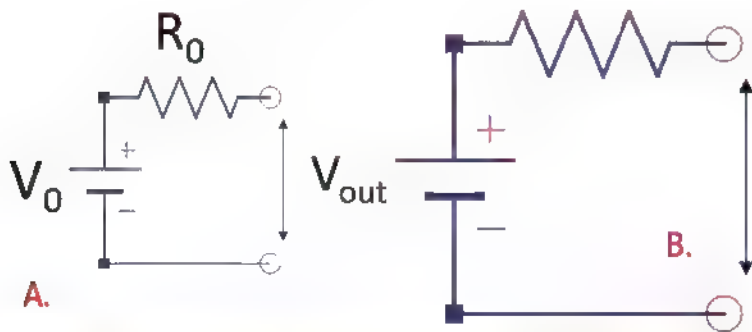
**FIGURE 6.** Measured DC and AC voltage response of the wall wart, nominally rated for 12V and 500 mA with a 25 ohm load. The performance is far from the rated spec.

in parallel have a resistance of  $50 / 2 = 25 \text{ ohms}$  – close enough to 24 ohms.

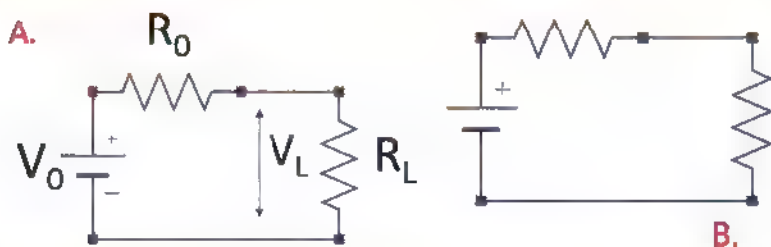
When this 25 ohm resistor was connected across the wall wart, the output voltage dropped considerably to 10.6V. **Figure 6** shows the measured voltage response of the wall wart.

The wall wart – with about 500 mA, its “rated” current – had a voltage of 10.6V with a peak-to-peak ripple of 0.89V. This is 8% peak-to-peak noise. Even though it was stamped 12V at 500 mA, its output voltage was only 10.6V with 8% peak-to-peak ripple noise. This gives you an idea of what to expect in some cases.





**FIGURE 7.** The equivalent circuit of all power sources showing the two important elements: an ideal DC voltage source and an internal series resistor.



**FIGURE 8.** Equivalent circuit of the power source with an external load resistor.

## The Output Impedance of Power Supplies

The output voltage dropped from 13.6V unloaded to 10.6V with the 25 ohm resistor. All power sources will drop in output voltage when they supply current. This is due to their internal series resistance.

The simplest way of modeling the properties of a power supply is as an ideal DC voltage source with a series resistance. **Figure 7** is the equivalent circuit model. In this model, as current flows out of the power supply, there is a voltage drop across the internal resistor. What we see is the output voltage after the IR drop across the resistor.

There isn't really a series resistor at the output of the power source; it just behaves this way. There is usually some other regulating circuitry like a bipolar transistor or MOSFET, or internal components which electrically look like a resistor. Knowing the value of the ideal DC source voltage and the equivalent internal resistance of the supply tells us the most important properties of the supply. Plus, they are easy to measure.

The DC voltage of the ideal source is just the voltage measured when the supply is unloaded. The output resistance can be calculated based on the voltage drop when connected to a known load. **Figure 8** shows the equivalent circuit when the output is connected to a load resistor.

With a little circuit analysis, the internal resistor,  $R_0$ ,

can be calculated from the unloaded voltage,  $V_0$ , the loaded voltage,  $V_L$ , and the load resistor,  $R_L$ , as:

$$R_0 = R_L \frac{(V_0 - V_L)}{V_L}$$

To measure these features of any supply, just measure the voltage of the supply unloaded, then attach a known resistive load (watch out for smoke) and measure the voltage loaded. After a little arithmetic, you have the internal DC voltage and the internal resistive output.

The output voltage with any current load,  $I_L$ , can then be calculated as:

$$V_L = V_0 - I_L R_0$$

For example, in this wall wart, the internal resistance is:

$$R_0 = R_L \frac{(V_0 - V_L)}{V_L} = 25\Omega \frac{(13.6 - 10.6)}{10.6} = 25\Omega \frac{(3.0)}{10.6} = 7.1\Omega$$

Its output voltage with any current load is:

$$V_L = 13.6V - I_L \times 7.1\Omega$$

Need a 12V source when it draws 500 mA? Don't use this wall wart. In fact, when it draws more than 225 mA, the voltage will be below 12V. Unless you have characterized your wall wart, take the rating stamped on its outside only as a guideline — not a hard and fast performance spec.

While the internal DC voltage and the internal source resistor are really important, they are not the whole story. The other important term is the AC ripple noise when loaded down. These three terms are the most important to characterize the quality of a power source:

- Internal DC Voltage
- Internal Series Resistor
- Peak-to-peak AC Ripple when Loaded

## Building an Arduino Based Automated Power Source Analyzer

Now that we know what to look for in a power source, we can automate the process of characterizing it and, along the way, test the quality of this simple equivalent circuit model. Just how well does a power source look like an ideal DC voltage and a series resistor?

Since I am enamored with the power of Arduinos, I decided to use one at the heart of my automated system. If I can change the current draw of the power source and measure the supply voltage as it draws current and the

## How Not to Generate Smoke

A resistor is really a component which is very efficient at turning electrical power into thermal power and getting hot. The power dissipation in a resistor can be estimated from:

$$P = \frac{V^2}{R} = I^2R = V \times I$$

If we measure voltage in volts, current in amps, and resistance in ohms, the power dissipated in the resistor is in watts. Resistor sizes are generally based on the power they can easily dissipate before getting too hot. Small physical size resistors have a small surface area and can't get rid of the heat fast enough before getting hot. Large body sizes have a large surface area and are more efficient at getting rid of the heat.

Large size axial lead resistors can easily dissipate one watt; the next smaller size is 1/2 watt, and the next smaller is 1/4 watt. The tiniest axial lead resistors are generally rated for only 1/8 watt. It's possible to get larger body size resistors rated for even higher power dissipation. **Figure A2** shows examples of resistors with six different power ratings. It's all about their size.

Generally, if the power source is under 15V, to dissipate less than 1/2 watt and not worry about the power consumption requires a resistor greater than:

$$R = \frac{V^2}{P} = \frac{15^2}{0.5} = 450 \text{ ohms}$$

If you use a resistor less than about 500 ohms, be sure to put in the numbers to check if power will be a concern. If it is, watch out for smoke or engineer a solution that can adequately dissipate the power.



**Figure A2.** Resistors with six different power ratings: 10 watts, five watts, one watt, 1/2 watt, 1/4 watt, and 1/8 watt.

current load at the same time, I can plot the I-V curve. This should look like a straight line, with an intercept at the DC voltage and a slope equal to the internal resistor. From this curve, I can fit these two terms.

I used a simple transistor follower circuit as a resistive load. The input voltage to it will control the current. I wanted to ramp the current up so I could measure a few different values from 0 to a maximum. So, how do I get a ramp voltage from an Arduino?

Even though the specs for an Arduino say it has analog output, this is not quite correct. There are six digital pins — usually designated with a little squiggly line (“~”) next to them — which can be set up as pulse width

## Resources

[www.digilentinc.com/Products/Detail.cfm?NavPath=2,842,1018&Prod=ANALOG-DISCOVERY&CFID=7824927&CFTOKEN=628690e39396761f-CE94CA68-5056-0201-02093BB386E8F629](http://www.digilentinc.com/Products/Detail.cfm?NavPath=2,842,1018&Prod=ANALOG-DISCOVERY&CFID=7824927&CFTOKEN=628690e39396761f-CE94CA68-5056-0201-02093BB386E8F629)

[www.sparkfun.com/products/298](http://www.sparkfun.com/products/298)

[www.sparkfun.com/products/250](http://www.sparkfun.com/products/250)

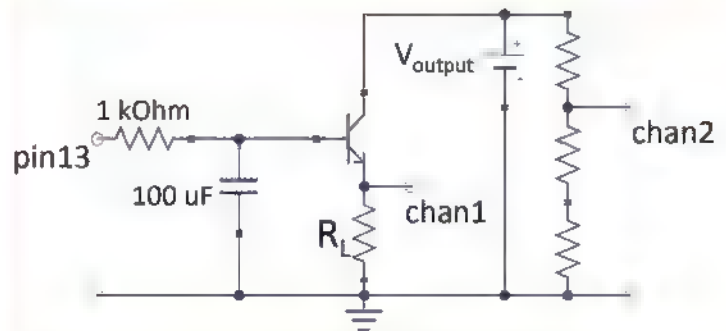
[www.sparkfun.com/products/9088](http://www.sparkfun.com/products/9088)

[www.dataq.com/products/di-145/](http://www.dataq.com/products/di-145/)

<http://labjack.com/u3>

[www.sparkfun.com/products/12757](http://www.sparkfun.com/products/12757)

[www.parallax.com/downloads/plx-daq](http://www.parallax.com/downloads/plx-daq)



**FIGURE 9.** Transistor follower circuit with an RC integrator.

modulated (PWM) signals. The duty cycle of a 500 Hz square wave is adjusted so the integrated value — or average value — can be changed.

However, to use these pins as an analog voltage — for example, to ramp up the voltage and the current load — would require a long integration time and they would still have some ripple on them.

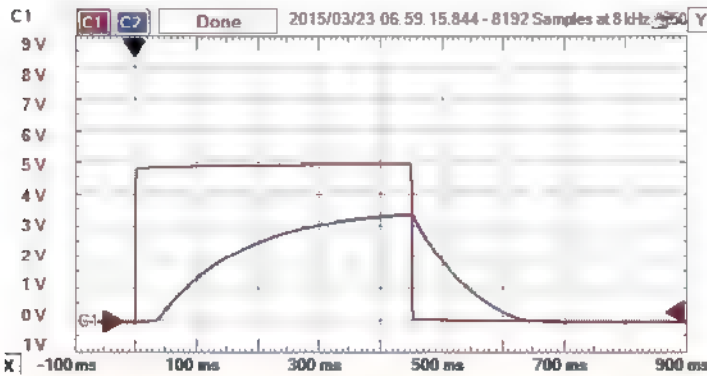
I decided to use another approach to generate a ramp signal. I know I can get a precisely controlled digital output signal which is basically a square wave pulse. I could use a simple RC filter to turn the digital signal into a ramp. The output of the digital pin would charge up a capacitor through a resistor.

This voltage would then control the current through the transistor and the load on the power supply. **Figure 9** shows this simple circuit.

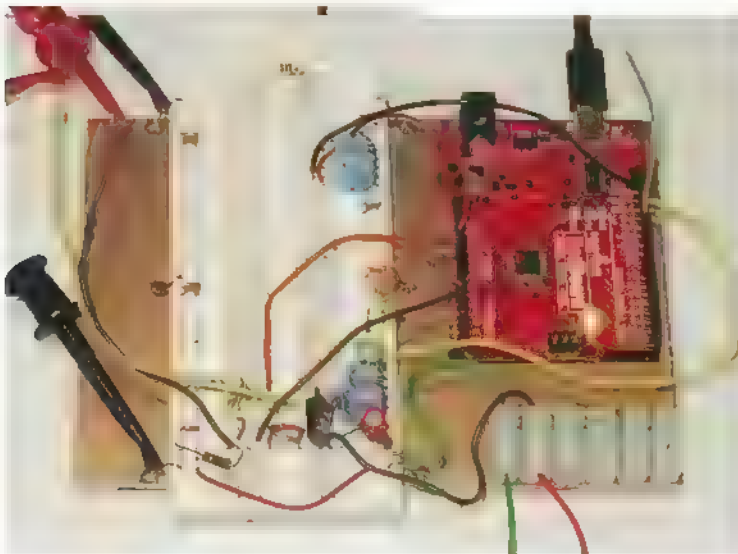
For this application, I used a 2N6040 — an NPN Darlington transistor. By adjusting the load resistor,  $R_L$ , I can get any range of current I want. Given the limited number of time constants I wanted to wait and the forward drop through the transistor, the maximum voltage I would see across the resistor is about 3.5V. With a 10 ohm load resistor, the current will range from 0V to  $3.5V/10 \text{ ohms} = 350 \text{ mA}$ .

I know I can set up an Arduino to sample the analog





**FIGURE 10.** Voltage on pin 13 of the Arduino and the voltage across the 10 ohm load resistor — a measure of the current from the power source. The square signal is the voltage on pin 13 and the ramping signal is the voltage across the 10 ohm resistor.



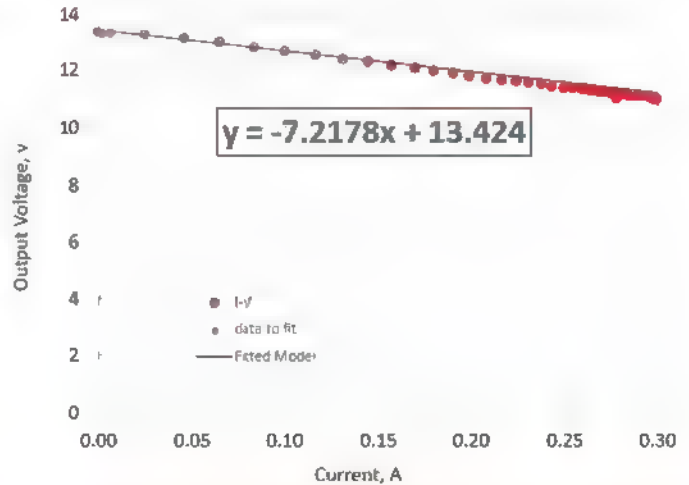
**FIGURE 11.** The final automated I-V system based on an Arduino with an integrating circuit and current load to a power source. The wires coming off from the bottom connect to the power supply to characterize.

inputs every 10 msec. For about 50 measured points during the ramp-up, this is an on-time of about 0.5 sec. As long as the current is on for just one second every so often, I am not worried about the power consumption in the resistor.

For an on-time of 0.5 sec, I needed an RC time constant of about 0.1 seconds. With a 1K ohm series resistor, this means an integrating capacitor of about 0.1 sec/1000 ohms = 100  $\mu$ F.

The voltage across the capacitor drives the base voltage, which then drives the current through the transistor. The voltage drop across the 10 ohm load resistor is a direct measure of the current through it.

**Figure 10** shows an example of the input voltage from digital pin 13 and the resulting voltage across the 10 ohm load resistor.



**FIGURE 12.** Measured I-V curve for the wall wart, showing the fitted values of the 13.4V open circuit load and 7.22 ohm output load impedance. This clearly shows how the output voltage of any power supply decreases with added current draw. The internal resistance is a good metric of how much voltage drop to expect.

## Choosing the Right Input Mode to a Scope

Most scopes have three input modes for a channel: 50 ohms, 1 megohm input, and AC coup ed. The 50 ohm input is most useful when measuring signals in the 100 MHz and above bandwidth. It will terminate the coax cable connecting the scope to the device-under-test and prevent the artifact of reflections in the cable.

The 1 megohm input is the general-purpose setting to look at signals from DC to about 100 MHz, depending on the nature of the probe. The AC coupled input puts a 100 nF capacitor in series with the 1 megohm. This means DC signals are blocked. The time constant is about 0.1 sec (100 nF x 1 megohm), so signals above about 5 Hz are passed through to the receiver pretty much unchanged.

This final system implemented with an Arduino Redboard from SparkFun (\$19.95) and a small protoboard is shown in **Figure 11**.

## Analyzing the I-V Curve in Excel

I wrote a simple sketch for the Arduino which sent a 0.5 sec digital pulse on pin 13 to drive current through the transistor from the power supply. I used two analog input pins to measure the voltage of the power supply and the current through it. *Chan1* was a direct measure of the current from the supply, while *chan2* was a scaled value of the supply output voltage.

Since many power sources are 15V or more, I used a voltage divider circuit to offer voltage drops from 1x, 0.5x, and 0.25x. This scaled the supply voltage to a range the

analog input pin could handle. I use enough voltage drop to keep the maximum voltage on *chan2* less than the 3.3V reference.

The sketch turns on pin 13, ramping the current from the power supply, and measures the supply current and the loaded voltage. These values are printed to the serial monitor.

I used PLX-DAQ to read the serial port and bring the data into a spreadsheet. The I-V data is plotted automatically as it comes in. I take the first five values to fit a linear trend line, and display the DC voltage and the slope which is the internal resistance. A steeper slope means a higher internal resistance.

When I click on the connect button in the PLX-DAQ control box, the Arduino is triggered to start the loop, the data is taken, printed to the serial port, read into Excel, and plotted — all automatically. **Figure 12** is an example of the measured I-V curve for this wall wart.

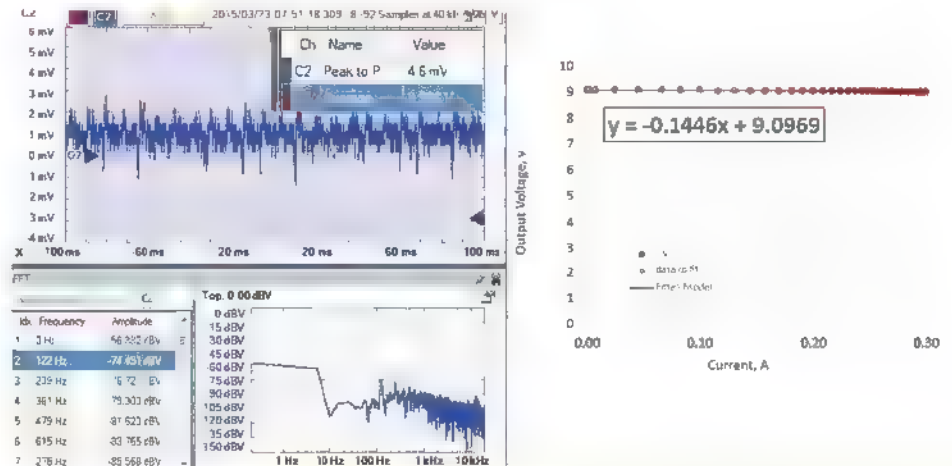
We can see that the model for a constant internal resistance is a pretty good one. The 7.22 ohm internal resistance measured with this automated system is very close to the 7.1 ohm resistance I measured manually.

With this automated system, we can now explore the features of different power sources.

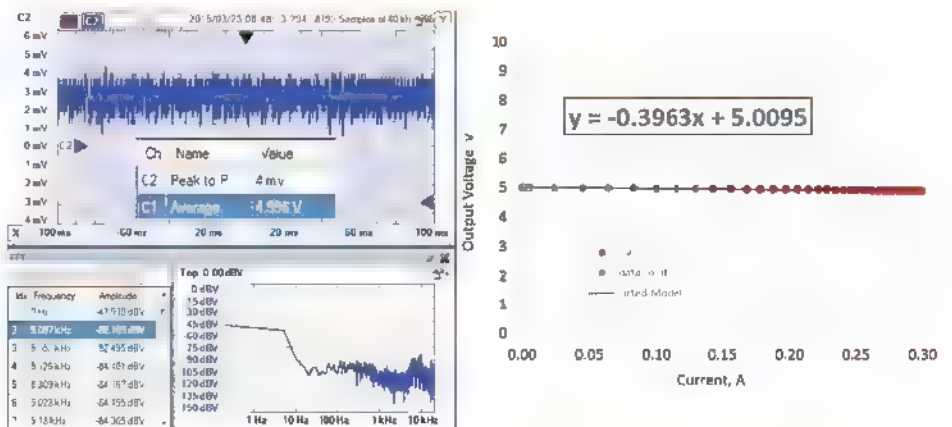
## A Survey of Power Sources

Not all wall warts are created equal. I routinely use a 9V wall wart from SparkFun all the time. It is rated for 9V at 650 mA. **Figure 13** shows the I-V curve and the AC ripple noise when loaded by 17 ohms, or drawing 530 mA. The peak-to-peak noise is remarkably low. The peak-to-peak voltage is 4.5 mV out of 9V, or less than 0.05% even loaded down. Its output impedance is 0.14 ohms.

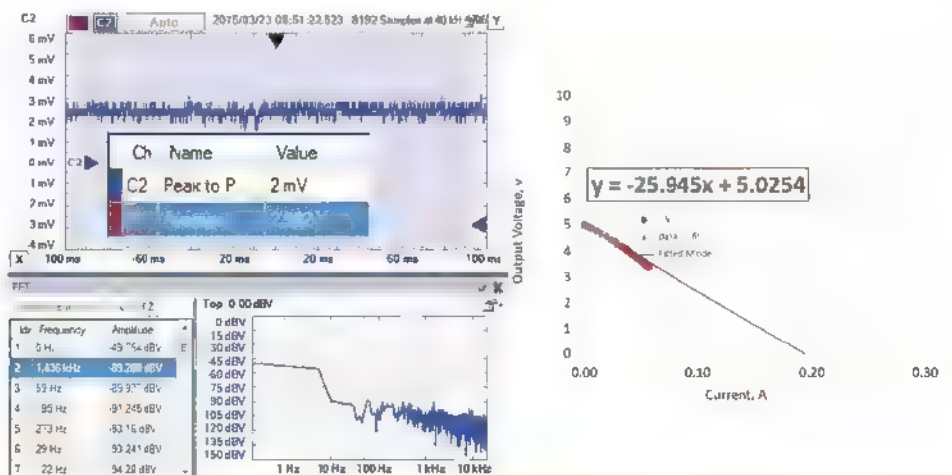
The USB port is often used to supply current. The measured I-V curve and AC ripple with a 42 ohm load drawing 120 mA is shown in **Figure 14**. It is a remarkably accurate 5V regulated supply with an output impedance



**FIGURE 13.** Characteristics of a 9V wall wart from SparkFun.



**FIGURE 14.** USB power performance.



**FIGURE 15.** Arduino digital output pin as a power supply.

of only 0.4 ohms. The AC ripple of the USB source I used was 4 mV peak-to-peak, with no discernable peak at 60 Hz or 120 Hz.

Sometimes one of the digital output pins of the Arduino is used to power a component. **Figure 15** shows its I-V curve and AC ripple noise when set to HIGH. The



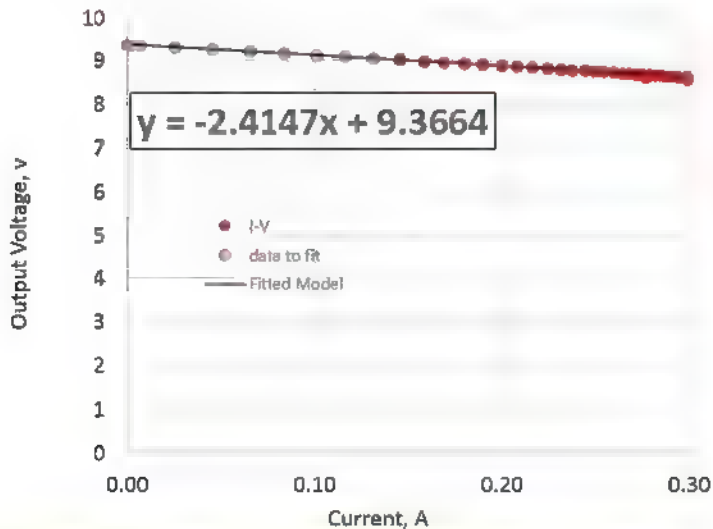


FIGURE 16. Measured output impedance of a 9V battery.

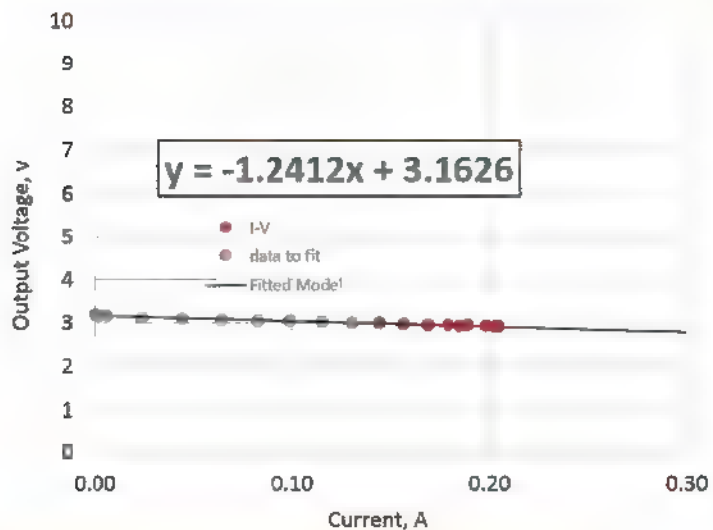


FIGURE 17. Measured output impedance of two AAA batteries in series.

output impedance of an Arduino pin is about 26 ohms. With a 40 ohm external resistor – or a total of 66 ohms – the current draw was  $5V/66 \text{ ohms} = 72 \text{ mA}$  current. The ripple noise was about 2 mV peak-to-peak, but only a very small peak at 60 Hz. The voltage amplitude of the 60 Hz sine wave component is  $10^{(-89.3/20)} = 34 \mu\text{V}$  out of 5V.

Many portable applications use batteries as the power sources. Their properties vary a lot depending on the age of the battery and the type of battery. The AC ripple noise is typically at the noise floor of what the scope can measure. The amount of AC ripple noise from a battery is not really about the battery, but about how well it is shielded from external noise sources, and what it is connected to in its application.

I surveyed a number of 9V batteries and found the typical output impedance to be about 2.4 ohms. An example is shown in Figure 16. However, one battery –

Eric Bogatin is a physicist on the faculty at CU, Boulder and Front Range Community College. He is also the Dean of the Signal Integrity Academy. He leads Arduino workshops at Tinkermill, the Longmont Hackerspace. His current interests are in simple, robust, and low cost electronics for high precision physics experiments. He can be reached at [eric@EricBogatin.com](mailto:eric@EricBogatin.com).

nominally also new – showed an output impedance of 8.5 ohms. This is just another example of the guideline: Buyer beware.

Two AAA batteries in series had an output resistance of 1.24 ohms. This implies the output impedance of one AAA battery would be about 0.62 ohms. Its IV curve is shown in Figure 17.

## Conclusion

Before you plug in that power source, try to answer four important questions:

1. What is the current requirement for your application?
2. Given the voltage you are going to use, what is the power dissipation you expect, and can you handle this?
3. What is the unloaded output voltage and source resistance of the power source you plan to use, and what will be the loaded output voltage? Is this adequate?
4. How much ripple noise is there on your power supply and is this okay in your applications?

With the simple measurement and analysis techniques offered here, you will be able to answer these questions and eliminate one common source of problems in your next project. **NV**

# All About Ferrite

Rings, beads, cores — hams recognize these instantly as the most common shapes of ferrite materials and soon you will, too!

## Ferrite — What is It?

Black, brittle, and heavy are all characteristics of ferrites — a family of ceramic materials made from oxides of metals like iron (Fe), zinc (Zn), manganese (Mn), and nickel (Ni). You have encountered ferrites as magnets and inductor cores and those odd lumps at the ends of cables. First created in 1930, they have since become important materials in the electronics and RF world. They have a high *permeability* (see the sidebar) but don't conduct electricity very well at all. However, these properties make them ideal for a variety of uses in electronics.

The ferrites made into permanent magnets are called *hard ferrites* which can be magnetized quite strongly. Hard ferrites are made from oxides of iron, cobalt (Co), barium (Ba), and strontium (Sr). They also have a high *coercivity* so they will stay magnetized and make great magnets.

As interesting as that may be, hard ferrites aren't all that useful in electronic circuits (except in motors), so we will leave them stuck to the refrigerator and move along.

If there are hard ferrites, there must also be *soft* ferrites. These are just as hard and brittle as the hard

ferrites, but have low coercivity and don't make good magnets. This is a good thing because their magnetization can switch direction easily and without much loss. Losses in iron build rapidly above audio frequencies, leaving ferrite as a good choice for inductor cores at the high frequencies encountered in switching power supplies and radio frequency (RF) circuits.

Ferrite's most important property is its permeability at different frequencies. But first — what *is* permeability? Applied to a membrane or filter, permeability means the ability of other substances to pass through it. Electromagnetic permeability is not so different: What passes through is magnetic energy, measured as *magnetic flux* which is a measure of the magnetic energy's density. Speaking in the electromagnetic sense, a material with high permeability not only allows magnetic energy to freely pass through, it has the ability to concentrate the available energy from a magnetic field.

A material's permeability is determined by the way its electrons respond to a magnetic field. If there are electrons available to line up their spins to reinforce the magnetic field, the material has a high permeability and will concentrate the field. Metals like iron, zinc, and cobalt have plenty of those electrons and so have high permeabilities. Non-metals like carbon or oxygen don't, and so their permeability is low. Permeability is usually given as a relative value, with air having a value of 1. Ferrites can have relative permeabilities from 10-20 to as high as 10,000!

You can observe the effect of permeability by winding a wire coil of 20 turns or so around a wooden pencil laid on a piece of paper.

Sprinkle some iron filings on the paper and run an amp or so of current through the wire. You'll see the filings respond to the magnetic field.

With the current off, pull out the pencil and replace it with a steel nail. Turn on the current and — wow! — the filings will flock to the ends of the nail showing the more concentrated magnetic field. That's the effect of permeability.

## It's All in the Mix

The name ferrite applies to many, many different types of material — each having properties tailored to various frequency ranges and applications. Because each different type of material is made from specific combinations of oxides, it is referred to as a *mix*; for example, a Type 43 mix or a Type 75 mix. Fair-Rite (see references) is a leading manufacturer of ferrites used in electronics, so the industry has more or less standardized on Fair Rite's mix designations. Keep in mind there is no guarantee that a Type 31 mix from company A is exactly the same as from company B. They may be close, but check the spec sheets if you have a critical application. Download the latest Fair-Rite catalog

## Knowing the Jargon

**Permeability ( $\mu$ ,  $\mu\mu$ )** — the ability of a material to store or focus magnetic energy; the degree to which a material becomes magnetized by an applied magnetic field.

**Coercivity** — the intensity of the magnetic field required to de-magnetize material after it has been magnetized as much as possible

**Saturation** — the point at which increases in inductor current do not cause a proportional increase of magnetic flux in the core.

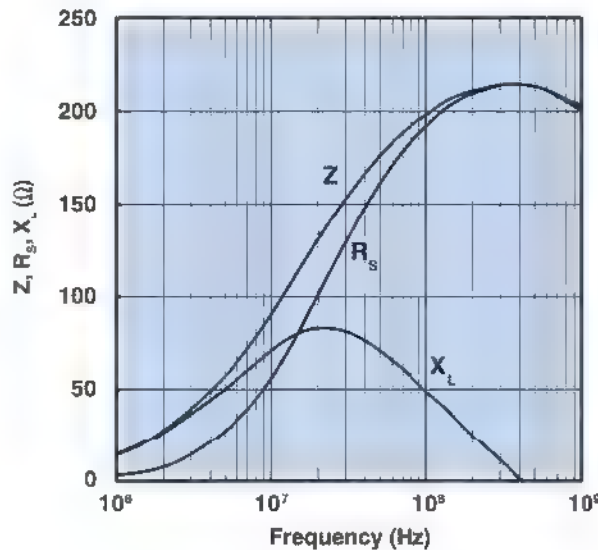


for a lot more good information.

**Figure 1** shows the relative impedance versus frequency behavior of the common Type 43 mix. If you stuck a wire through a Type 43 ferrite bead and measured the resistance, reactance, and impedance of that wire, whether you had just the one turn or multiple turns, the graphs would look like this. The equivalent circuit for the wire through the bead looks like **Figure 2**. (The reactance symbol has a square shape to let the reader know it is a property of the ferrite material.)

You can see that a ferrite core makes the inductor look like different components at different frequencies. At low frequencies around 1 MHz or less, the impedance is dominated by the reactance  $X_L$ , so the ferrite is storing and releasing energy just like a regular inductor. The resistive part of the impedance,  $R_s$ , rises rapidly and exceeds the reactance just below 2 MHz. The ferrite core inductor looks more and more like a resistor throughout the VHF region, with a peak impedance ( $Z$ ) of about 210  $\Omega$  around 300 MHz

In the inductive region, each ferrite mix has a different inductance-per-turn constant,  $A_L$ , which is given in units of mH or  $\mu$ H per 100 or 1,000 turns. The formula may also vary with the shape of the ferrite core on which the inductor is wound. Be sure you use the value of  $A_L$  and



Impedance, reactance, and resistance vs. frequency for a ferrite core in 43 material

**FIGURE 1.** An inductor with a ferrite core has properties that change with frequency. The type of ferrite material is chosen to give the desired performance over the intended frequency range.

*Photo courtesy of the Fair-Rite Corporation.*



**FIGURE 2.** A simple equivalent circuit for inductors with ferrite cores.  $jX_s$  represents the lossless inductive reactance and  $R_s$  represents resistive loss. Both  $jX_s$  and  $R_s$  change with frequency.

the formula given by the ferrite's vendor to calculate the number of turns or you may be greatly surprised!

Note that each pass of a wire or cable through a ferrite counts as a turn. It doesn't matter if a complete loop is made, just that the path goes

through the core.

Another way of looking at the ferrite's frequency-dependent characteristics is to look at its permeability versus frequency. We get surprises here, too!

There are two types of permeability shown in this graph. One type of permeability,  $\mu'_s$ , represents ordinary inductance,  $L_s$  — the material's ability to store and return magnetic energy to the electronic circuit. The other type of permeability,  $\mu''_s$ , describes the losses in the ferrite, just like resistance,  $R_s$ , in the ferrite equivalent circuit. You can see for Type 43 material,  $\mu'_s$  dominates at low frequencies just like  $X_L$  dominates in **Figure 1**. At higher frequencies,  $\mu''_s$  is the dominant factor. The composite permeability is

actually a complex value:  $\mu = \mu'_s + j\mu''_s$

## Choosing a Mix

Fair-Rite makes quite a number of mixes, and **Table 1** shows typical uses and permeability of several common ones. There are two primary uses. Fair-Rite refers to them as "inductive" and "EMI suppression" products. (EMI stands for *electromagnetic interference*.)

Not surprisingly, a mix optimized for use as an inductive product is designed to store and release magnetic energy with low losses. This would be the kind of material you would use in a switchmode power

**Table 1 — Common Ferrite Mixes and Permeabilities.**

Mix	Permeability ( $\mu$ )	Inductive Range (MHz)	EMI Range (MHz)
31	1500		1-300
61	125	<100	200-2000
43	850	<10	25-300
77	2000	<3	
75	5000	<0.75	

Note — Data from Fair-Rite catalog, 16th edition

## How is Ferrite Made

Mixes of metal oxide powders and various binding agents are placed into a mold, heated, and squeezed until the ceramic is created. This process is called sintering. If the ferrite is to be made into a magnet, the hot ferrite may be magnetized so that when it cools, the magnetism is retained. Ferrites used in electronics are not magnetized, however.

supply inductor. When the switch in the supply's circuit is on, current flows through the inductor, "charging" it with magnetic energy. When the switch is turned off, the stored energy is returned to the circuit in the form of current.

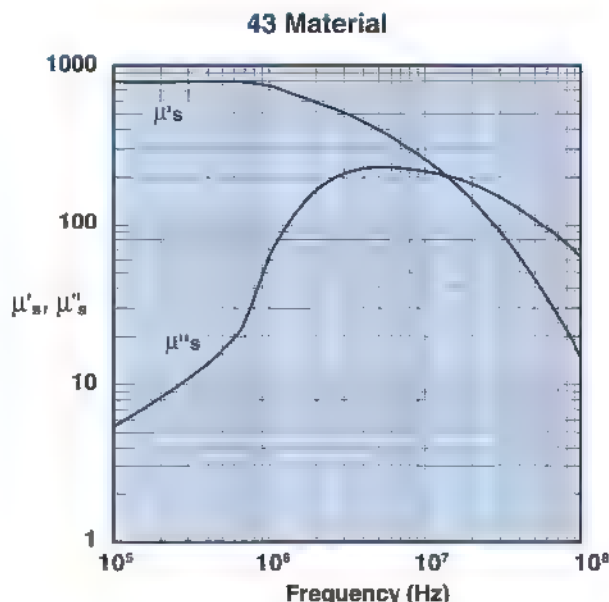
In this application, you want  $\mu'_s$  to be the dominant type of permeability at the switching frequency. Another inductive application in radio would be as an output transformer in an amplifier circuit or as an impedance transformer for matching the impedances of an antenna and a feed line.

On the other hand, why would you want a high loss component? In order to suppress or get rid of radio frequency energy, you don't want that! There are two reasons for wanting to suppress EMI — keeping RF out of something it shouldn't get into and keeping RF inside something it shouldn't get out of. RF is like a cat — always on the wrong side of the door.

Hams experience both quite frequently. A strong transmitted signal can easily be picked up by a cable and conveyed to a susceptible piece of equipment.

Conversely, microprocessor or network gear or a cheaply made switching supply can leak RF signals through cable connections that are then radiated and cause interference themselves. Both situations are common.

We'll actually cover the ways ferrite is used to manage RF in the next Ham's Wireless Workbench but, for now, let's consider why a lossy material is useful. Let's say you have a shielded audio cable picking up a strong local signal. — a



Complex Permeability vs. Frequency Measured on a 17/10/6mm toroid using the HP 4284A and the HP 4291A.

**FIGURE 3.** The permeability of ferrite has two components:  $\mu'_s$  represents the lossless permeability for energy storage, while  $\mu''_s$  represents the lossy permeability. Photo courtesy of the Fair-Rite Corporation.

ham or CB transmission, a paging or dispatch transmitter, or even a nearby TV or FM station's broadcast.

Certainly you could use the inductive material — slipping a ring of ferrite over the cable makes the outer surface of the cable into a small inductance which places some

reactance ( $X_L = 2 \pi f L$ ) in the path of the unwanted RF current. (The inside of the cable is largely unaffected.)

So far, so good. But that doesn't really get rid of the unwanted current; it just redirects it somewhere else. If the cable's outer surface has a capacitive reactance, adding the inductive reactance in series with it might even reduce the overall reactance and let more current flow. Better by far to make the impedance resistive which dissipates the energy of the unwanted signal as heat. Thus, for EMI applications in which you are trying to get rid of RF energy entirely, a lossy material is preferable.

Since ferrite's permeability changes with frequency, you might find a mix with one frequency range at which it's a great material for inductive energy storage, and another frequency range at which it makes a great energy dissipater. Just don't get the two confused!

Type 43 is one such material. Below 10 MHz, Fair-Rite recommends it for low power inductive uses while from 25-300 MHz, it's a dandy EMI suppression material. In fact, most of the cores you find molded on data and audio/video cables use this mix.

## Telling a Ferrite by Eye

Ferrite cores come in three basic types like what you can see in **Figure 4**: toroids (rings); beads (cylindrical sleeves); and clamp-ons that snap together around a cable. Toroids have the advantage of there being no ends to the core, so the inductor's magnetic field stays almost completely within the core — good for



**FIGURE 4.** Ferrite cores are available in several common sizes and shapes, ranging from tiny sleeve beads that slip over an individual wire to giant clamp-on cores that can be placed on power cables. Photo courtesy of the ARRL.



## Watching and Listening

Podcasts and videos are great ways to discover and explore technical topics — many are listed on the ARRL's Tech Portal page at [arrrl.org/tech-portal](http://arrrl.org/tech-portal). Circuit builders of any stripe will enjoy the Soldersmoke podcasts and videos archived at [www.soldersmoke.com](http://www.soldersmoke.com) (check the blog link for the latest programs). Randy Hall K7AGE has a YouTube channel ([www.youtube.com/user/K7AGE/videos](http://www.youtube.com/user/K7AGE/videos)) covering lots of technical and operating topics, plus interviews galore. Check out his visit to the WLW 500,000 watt transmitter site — those are the Big Dogs!

# High-Power Analog Autotuner



## HF-AUTO



The HF-AUTO is a microprocessor controlled fully automatic stand-alone tuner with a power rating of 5 Watts to 1800 Watts that will work with any transmitter built from the 1940s to the present. HF Bands: 160m to 6m. Three antenna outputs: SO239 coax. Dimensions: 12 5" W x 6 5" H x 16.5" D. Weight: 25 lbs. (11.4 kg).

**PALSTAR**

Customer Support: 1-800-773-7931  
Fax: (937) 773-8003

[www.palstar.com](http://www.palstar.com)

avoiding unwanted coupling and leaking of signals.

Beads are available from miniscule (they are even available in SMT packages) to fist-sized, and have to be slipped over a wire or cable before connectors are put on or attached. Clamp-ons are useful for large cables such as coaxial cable and power cords, and when a connector is already installed so a bead or toroid can't be used.

So, here you are at the robotics or hamfest flea market and — what a deal! — a vendor has a big box of toroid cores labeled "Ferrite cores - 4/\$1." Sorry. Walk on by. Unless they are marked with the mix used to make the cores, you have no idea how they will behave. You don't really even know if they are ferrite or powdered iron. Powdered iron cores are often painted certain colors if they are from Micrometals or Amidon, but unless you know the source don't trust the color.

Neither is it a good idea to take that box of cores home and see if you can figure out what mix they are by winding a few turns on them and measuring the inductance. Many mixes can give approximately the same result, and inductance won't give you any clue as to the behavior at different frequencies. Heed my advice and pass up the surplus cores. Although, they do make nice paperweights.

This is a good reason for clearly labeling the cores you do buy for

## Ferrimagnetism versus Ferromagnetism

What we usually think of as magnetic materials — zinc, iron, nickel — are ferromagnetic. These elements have unpaired electrons in the outer shell of atoms that line up with each other and stay lined up. They retain their magnetization once a field has been applied. Ferrimagnetic materials also have unpaired electrons but unlike the ferromagnetic materials, their electrons tend to line up opposing each other, so the resulting magnetic field is almost completely canceled out.

## References

Fair-Rite

[www.fair-rite.com](http://www.fair-rite.com)

Ferroxcube (Yageo)

[www.yageo.com](http://www.yageo.com)

Magnetics

[www.mag-inc.com](http://www.mag-inc.com)

Micrometals

[www.micrometals.com](http://www.micrometals.com)

Amidon Company

[www.amidoncorp.com](http://www.amidoncorp.com)

future reference. A piece of colored tape with the manufacturer and part number will identify the part exactly — no guessing. Or, write on the core with a pencil or permanent marker. Another good idea for your ferrites is to not throw them all in a box unprotected. Remember that ferrites are brittle — they chip and crack easily. Once broken, the core is probably unuseable. Put the core in a plastic bag or paper envelope for a little buffer against its fellow cores.

## Ferrite — the EMI Fighter

In the next installment of the Ham's Wireless Workbench, I'll show how ferrites are applied in the never-ending battle against EMI from, to,

### Ferrite or Powdered Iron?

Powdered iron cores have a much lower permeability than ferrite, but can also handle the higher levels of magnetic flux at high power without saturating. Saturating distorts the applied waveform, creates spurious harmonics and other unwanted signals, and can seriously over-stress a transistor. For transmitting and other high power applications, powdered-iron or powdered-permalloy cores are the usual choice. The free paper, "A Critical Companion of Ferrites with Other Magnetic Materials" ([www.mag-inc.com/File%20Library/Product%20Literature/Ferrite%20Literature/cg-01.pdf](http://www.mag-inc.com/File%20Library/Product%20Literature/Ferrite%20Literature/cg-01.pdf)) — an excellent overview of different magnetic materials from Magnetics — is a must-read if you want to know the details.

and by electronic gadgets of all sorts. We'll also cover a few other tricks

the hams have learned that you can apply as well! **NV**

## Pi-RAQ

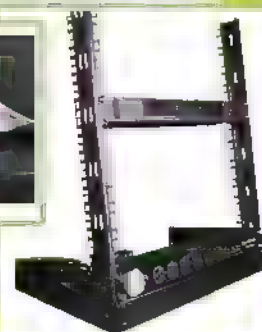
*An open source Raspberry Pi Based 1U Rack Mount Internet Appliance With the World's Only 10" x 1" TFT*

With the innovation of the Pi Raq and existing Earth.CD 10x1 display, Rack mount equipment, such as power controllers, audio/video switchers, network hubs and routers, can now have an easily programmable front panel with a color TFT LCD



### Features at a glance:

- 1- 10" x 1" Color TFT
- 2- 1024 x 100 pixels
- 3- 250 Nts
- 4- 7 Function Tact and Scroll Wheel
- 5- 9-20V Input Power Supply
- 6- 10x1 LCD Adapter Board
- 7- 4 USB Ports



Contact Us: [sales@earthlcd.com](mailto:sales@earthlcd.com)  
[www.EarthLCD.com](http://www.EarthLCD.com)

949-248-2333 Direct

## Make up to \$100 an Hour or More!



# Be an FCC LICENSED ELECTRONIC TECHNICIAN!

**Get your "FCC Commercial License" with our proven Home-Study Course!**

- No costly school. No classes to attend.
- Learn at home. Low cost!
- No previous experience needed!
- **GUARANTEED PASS!** You get your FCC License or money refunded!



Your "ticket" to thousands of high paying jobs in Radio-TV, Communications, Avionics, Radar, Maritime and more.

Call for **FREE** info kit: **800-877-8433** ext. 109

or, visit: [www.LicenseTraining.com](http://www.LicenseTraining.com)

**COMMAND PRODUCTIONS • FCC License Training**  
Industrial Center, 480 Gate Five Rd., PO Box 3000, Sausalito, CA 94966-3000





# ELECTRONET

**CORIDIUM**  
 Floating point **BASIC** for  
**ARM** controllers from \$5.00  
[www.coridium.us](http://www.coridium.us)

**ALL ELECTRONICS**  
 Electronic Parts  
 and Supplies.  
[www.allelectronics.com](http://www.allelectronics.com)  
 Free 96 page catalog 1-800-826-5432

For the ElectroNet  
 online, go to  
[www.nutsvolts.com](http://www.nutsvolts.com)  
 click **Electro-Net**

**USB** Add USB to your next project--  
 It's easier than you might think! **DLP Design**  
 USB-FIFO • USB-UART • USB/Microcontroller Boards  
 RFID Readers • Design/Manufacturing Services Available  
*Absolutely NO driver software development required!*  
[www.dlpdesign.com](http://www.dlpdesign.com)

**INVEST in your BOT!**



**INVEST in HITEC**

13115 Parne Street, Newark, CA 94664 958-746-1048 [www.hitecprod.com](http://www.hitecprod.com)

**HOBBY ENGINEERING**  
 Kits, Parts and Supplies  
[www.HobbyEngineering.com](http://www.HobbyEngineering.com)

PCB, PCBA and More! **Myro** Low cost High Quality  
[www.myropcb.com](http://www.myropcb.com)  
 1-888-PCB-MYRO

**Ironwood ELECTRONICS** **Burn-In & Test Sockets**  
 0.4mm to 1.27mm  
 Industry's Smallest Footprint  
 1-800-10-10208  
[www.ironwoodelectronics.com](http://www.ironwoodelectronics.com)

**superbrightleds.com**  
 COMPONENT LEDs • LED BULBS • LED ACCENT LIGHTS



**ramsey** **GET THE NUTS-VOLTS DISCOUNT!**  
 Mention or enter coupon code **NV10M2142**  
 and receive 10% off your order!  
[www.ramseykits.com](http://www.ramseykits.com)  
 AM/FM Broadcasters • Hobby Kits  
 Learning Kits, Test Equipment  
 AND LOTS OF NEAT STUFF!



Wanna learn more about electronics?  
[www.nutsvolts.com](http://www.nutsvolts.com)

**MOBILE APP NOW AVAILABLE!**

**SERVO**  
 Download NOW On Your  
 Favorite Mobile Device!

**FOR ROBOT BUILDERS**

**IOS • ANDROID • KINDLE FIRE**


[www.servomagazine.com](http://www.servomagazine.com)



**Did You Know Preferred  
 Subscribers get access to all the  
 digital back issues of  
 Nuts & Volts for free?**

Call for details  
**1-877-525-2539**

# Smart Mice READ SERVO MAGAZINE



*Brought to you  
by the publishers  
of Nuts & Volts!*

## Shouldn't You?

*For Robot Lovers, Tech Junkies, And MechHeads*

**SUBSCRIBE NOW!**  
12 issues for \$26.95

[www.servomagazine.com](http://www.servomagazine.com) or  
call toll free 1-877-525-2538

Use Promo code G57SNV



# Building a Multipurpose Communications Board

In this edition of *Design Cycle*, we're going to do as the title of this column says: walk through a complete design cycle from thought processes to the last solder joint. The project consists of designing and building a multipurpose communications node based on a 32-bit PIC microcontroller. The entire project will be realized in the ExpressPCB format. Using ExpressPCB allows the schematic diagram and printed circuit board (PCB) layout to be easily distributed to you. You can then take the project's engineering graphics and duplicate/modify the efforts put forth in the text of this column. There's lots of work yet to do. So, let's get started.

## The Design Goals

Size is not important as the project will not be designed to fit into a specific enclosure or exist in a restricted space. With that, we will start with the intent of being able to place our components within the confines of a standard 3.8 x 2.5 inch two-layer ExpressPCB printed circuit board.

From a work effort point of view, a four-layer board would make things easier because then we can simply drop power and ground connections anywhere they are most convenient. However, that design luxury comes at a higher board price. So, we will fly coach and put our design down on a two-layer PCB.

In that we are designing a communications board, we may be forced to make a decision on the PCB layers based on the requirements of our communications equipment. We will try to avoid going to a four-layer board. So, that means we may be limited in our choice of radio equipment. In the end, it will be a price versus performance balancing act.

## The Microcontroller

In this case, selecting a microcontroller is dictated more by what we *don't* need. For instance, we don't need a ton of general-purpose I/O. However, we will probably need multiple USARTs and at least one SPI portal. Ease of use is also high on our list. Believe it or not, the Microchip PIC32MX microcontrollers are as easy to use as many of the smaller less complex microcontrollers.

This relative ease of use is made possible by the way the PIC32MXs allow the programmer to manipulate the general-purpose I/O pins. The PIC32MX microcontrollers allow single-cycle set, clear, and invert I/O operations. For instance, to set the most significant bit of 16-bit I/O port X, we simply code `LATXSET = 0x8000`. To clear that same bit, we issue the mnemonic `LATXCLR = 0x8000`.

Toggling the most significant bit is just as easy as coding `LATXINV = 0x8000`. Other desirable features of the PIC32MX are an onboard RTCC (Real Time Clock Calendar), multiple precision timers, and native USB capability.

I have selected the PIC32MX575F512H. This PIC is a 64-pin device that is pin-compatible with the Ethernet-capable PIC32MX675F512H. If we ever want to morph this design into an Ethernet-capable node, we can simply drop in the PIC32MX695, preserving the basic layout work we already have.

Choosing the PIC32MX575F512H gives us 64 KB of SRAM and 512 KB of program Flash. If we wish to employ a bootloader, there's 12 KB of boot Flash at our disposal. The PIC32MX575F512H can support up to six UARTs, mixed in with support for up to four SPI portals. A USB portal is supported that can be configured as a host, device, or OTG (On-The-Go) port.

I have other plans for USB support, so we most likely won't deploy the PIC32MX575F512H's native USB capabilities. However, we will take advantage of its five timer modules. At this point, there are no requirements for an SPI portal. That leaves us with the option of deploying the maximum number of UART portals native

to the PIC32MX575F512H.

Our initial design will include an 8 MHz clock crystal that via the PIC32MX575F512H's internal PLL will enable 80 MHz operation of its CPU. We will also design in a 32.768 kHz clock crystal to drive the PIC32MX575F512H's internal RTCC module.

The RTCC provides "human time" clocking for timing of events that occur in time-of-day intervals (hours-minutes-seconds). Who knows. That might come in handy. I often use the one second alarm capability of the RTCC to blink an "I'm alive" LED in addition to providing timing intervals based on seconds.

## Mass Storage

Yep. Mass storage. It would be nice to be able to store whatever we find necessary and keep it stored until we need it. The PIC32MX575F512H is a very powerful computing device. However, when the lights go out, so does everything else in the "house." We're not designing a battery-operated device. So, we'll have to call upon a storage device that has become very popular with embedded designers: the microSD card.

Instead of attaching our microSD card to one of the PIC's SPI portals and coding a driver, we'll take a different road. We're going to relieve the PIC32MX575F512H of the burden of executing the low level code that drives a typical microSD card. We're also going to give ourselves a break by not having to adapt and integrate the base Microchip FAT drivers.

You may find this amusing. We're giving control of the microSD card to an Atmel microcontroller.

Our microSD subsystem is based on an open source project called OpenLog. You can buy an OpenLog from SparkFun. Or, as long as you acknowledge the open source folks behind it, you can also lay one down on a PCB of your choice. OpenLog was originally intended as an "instant gratification" data logger.

The idea behind OpenLog is to power-up and just log data. You can get all of the OpenLog scoop you need from the SparkFun site and GitHub.

The OpenLog circuitry attaches to our PIC32MX575F512H via its UART ports. Instead of just logging data, we're going to configure our instance of OpenLog to act as an on-demand microSD storage device.

By simply exchanging microSD cards, we can create a mass storage device that can store from 1 GB to 64 GB. On our command, every bit that the OpenLog processes and stores will flow through one of the PIC32MX575F512H's UARTs.

The advantage of using a microSD card instead of a nonvolatile device such as an EEPROM is that we can easily pass the data to another embedded device or a PC

by simply plugging and unplugging the microSD card.

## USB

We've already poo-pooed the PIC32MX575F512H's on-chip USB module. Our device is not designed to be a dedicated USB device or USB host. So, to utilize our communications node as an adhoc USB portal, we must "install" our own USB hardware.

Our design will use the USB portal just like it was an RS-232 port. To keep it as simple and as robust as possible, we'll go with a proven method based on the FTDI FT232RL. Again, we're saving finger time as there is no special driver code we have to write to utilize the FT232RL.

## RS-232

Despite the movement towards killing good old RS-232, it's still standing. This design wouldn't be complete without a regulation RS-232 interface. With that, we'll design in a three-wire RS-232 serial port with an optional RTS signal. Our RS-232 portal will be based on an ST3232 RS-232 interface IC.

## Radio

We want to get as much functionality out of our design as possible. The popularity of the XBee 802.15.4 radio module helps us in this respect. We can design in a

## RESOURCES

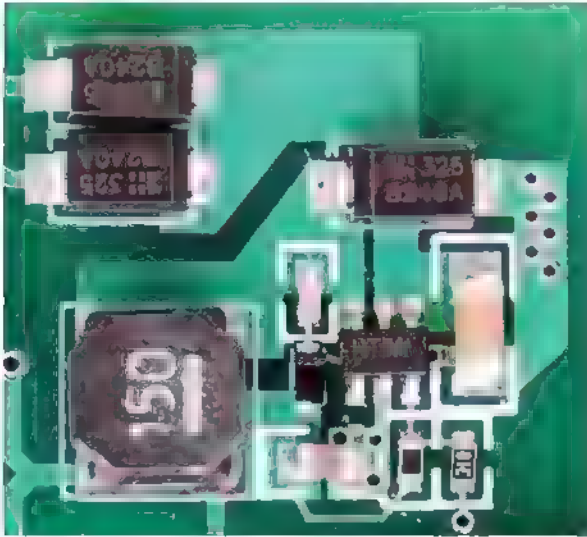
**Microchip**  
 PIC32MX575F512H  
 MPLABX  
 XC32  
 PICKit3  
[www.microchip.com](http://www.microchip.com)

**ExpressPCB**  
 Printed Circuit Boards  
 Printed Circuit Board Design Tools  
[www.expresspcb.com](http://www.expresspcb.com)

**SparkFun**  
 OpenLog  
[www.sparkfun.com](http://www.sparkfun.com)

**CCS**  
 PIC C Compiler  
[www.ccsinfo.com](http://www.ccsinfo.com)





■ Photo 1. The PCB layout doesn't give you a good visual perception of the switching power supply. This overhead shot should bring things into perspective.

standard XBee dual 10-pin footprint which will allow us to communicate using 802.15.4 or Wi-Fi.

Depending on the application needs, we can deploy a

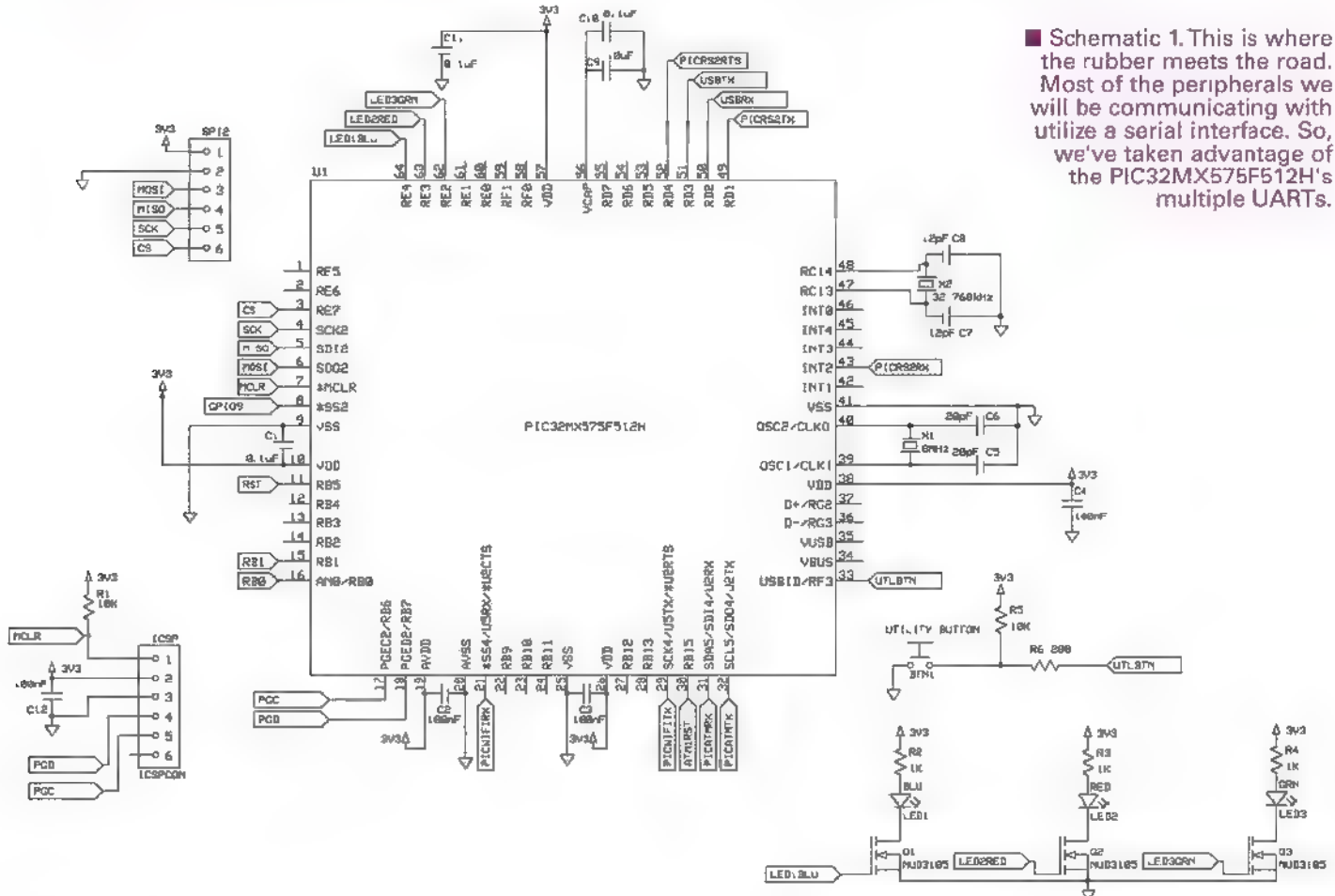
standard XBee radio, an ACKme Wi-Fi radio, or a Roving Networks WiFly radio. I've seen other XBee footprint-compatible radios that should work in this environment, as well. The idea is to be able to plug and play these variants per our application requirements.

## Power

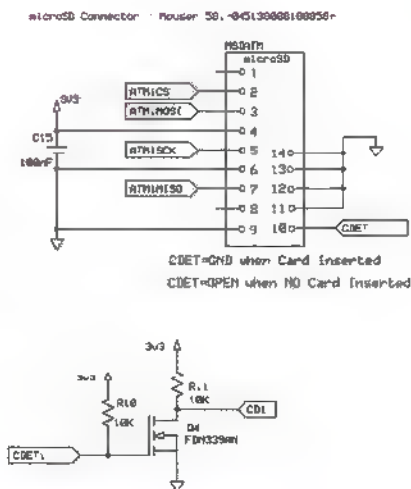
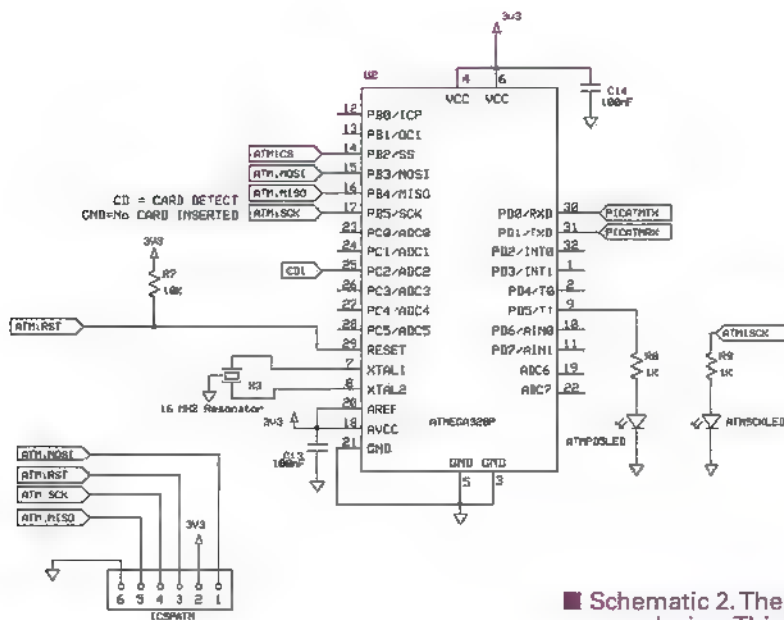
Let's take the road less traveled. Instead of designing in a standard LDO linear voltage regulator, we'll design and install a switching regulator. The advantages of our switching regulator design include a small footprint, minimal heatsink area, and a wide input voltage range.

Our switcher will be based on the Microchip MCP16301T which can provide +3.3 VDC at up to 600 mA. The input voltage supplied to the MCP16301T can range from 4 to 30 VDC.

The MCP16301T's wide input voltage operational range allows our multipurpose communications node design to be easily powered by a cheap 9 VDC unregulated wall wart. This is flight tested hardware. I've used this power supply design for a number of other projects. I've included an aerial view of this power supply circuit for your enjoyment in **Photo 1**.



■ Schematic 1. This is where the rubber meets the road. Most of the peripherals we will be communicating with utilize a serial interface. So, we've taken advantage of the PIC32MX575F512H's multiple UARTs.



■ Schematic 2. The FTDI FT232RL is carrying the USB load in our design. This utility USB portal can be used for debugging, program control, and fast data transfers.

## Eye Candy

The PIC32MX575F512H is a wonderful microcontroller. However, it can't do everything. By that, I mean it can't power and provide computational support for everything you want to hang on its I/O pins. That applies particularly to LEDs.

To avoid exceeding the PIC32MX575F512H's maximum current limit on its I/O pins, we can buffer our LED indicators with NUD3105 MOSFETs. Doing this passes the LED current load to the NUD3105 devices. The PIC32MX575F512H sees the MOSFETs as low current digital I/O devices instead of high current LEDs.

With only three user-defined LEDs, our design won't get anywhere close to the I/O pin current limit maximum. The inclusion of the MOSFETs is more of a personal preference here. We could easily drive the three LEDs directly from the PIC32's I/O pins.

We will also design in a utility momentary normally open pushbutton switch. A pushbutton switch could come in handy as a bootloader start signal. The switch could also be helpful with debugging or play a role in an application.

## Design Details – Microcontroller

The microcontroller hardware details are outlined graphically in **Schematic 1**. The major players are the UART ports. UART1 is dedicated to the FT232RL USB UART IC, whose supporting circuitry is outlined in **Schematic 2**. A firmware implemented interrupt-driven data buffer stands behind the UART1 receive pin. The receive interrupt mechanism makes sure that all of the

incoming data flowing between our communications board and the external host USB device is captured and accounted for.

Debugging is the main idea behind designing in the FTDI device. I see it mainly used to interface to the CCS C serial I/O tool that comes with the CCS PIC C compiler. External devices or programs could also use this USB portal to issue application-specific commands or transfer packets of data.

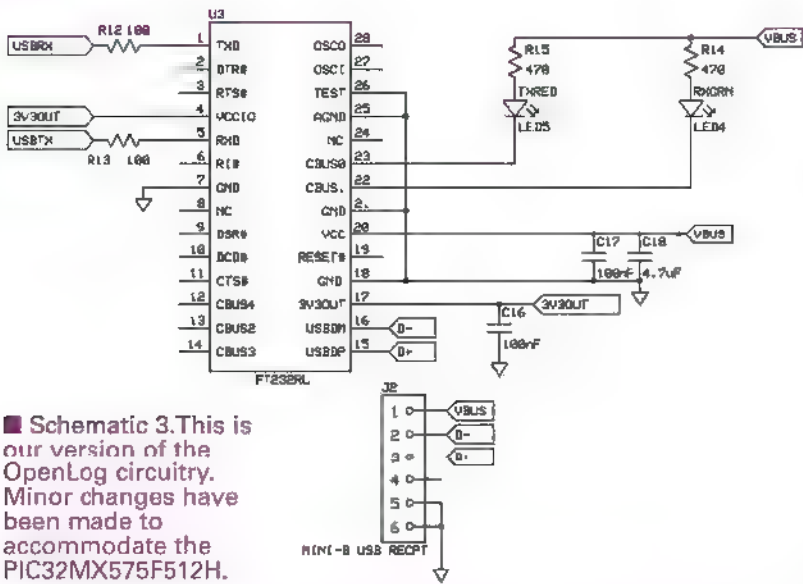
UART2 is responsible for communicating with the ATmega328P that drives the microSD card. File system commands and application data will flow between the PIC32MX575F512H's UART2 and the ATmega328P's UART. The only coding we have to perform to realize this interface is setting up the PIC32MX575F512H's UART2.

Assigning the RS-232 portal duty to the PIC32's UART4 preserves SPI portal 2. SPI portal 2 shares pins with the PIC32MX575F512H's UART3. This is a win-win situation in that we can accommodate an additional UART interface or support an SPI portal. For convenience, we'll pin out the SPI portal to a DF13 header.

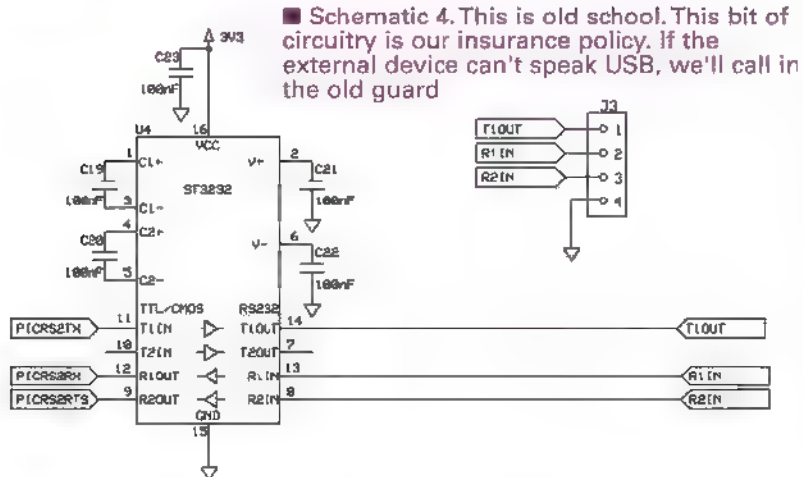
Talking to the multi-unit radio socket is the responsibility of the PIC32MX575F512H's UART5. The XBee-type radio modules share a common three-wire serial interface. So, a simple serial communications interface is all that is required. The radio data interface includes a common RESET I/O line and an additional adhoc mode I/O signal that is specific to the WiFly module.

All of the LEDs that are under the control of the PIC32's I/O pins are driven by the user's application code. The same is true for the pushbutton switch.





■ Schematic 3. This is our version of the OpenLog circuitry. Minor changes have been made to accommodate the PIC32MX575F512H.



■ Schematic 4. This is old school. This bit of circuitry is our insurance policy. If the external device can't speak USB, we'll call in the old guard.

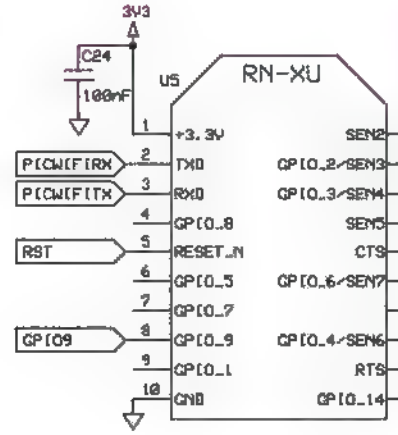
The PIC32MX575F512H is programmed in the standard Microchip manner. My choice of PIC programming tools includes the PICKit3, XC32, and MPLABX. I've selected a DF13 1.25 mm-pitch connector for the PICKit3 interface job. You can follow your own heart as far as this connector and your choice of programming tools is concerned.

### Design Details – FTDI Interface

This is a stock design. The FT232RL does all of the heavy lifting without any assistance from the PIC32MX575F512H. As you can see in Schematic 2, the FT232RL supplies its own eye candy.

### Design Details – microSD Module

Our version of the OpenLog microSD hardware is drawn up in Schematic 3. The real magic resides in the open source ATmega328P firmware. As you can see, the



■ Schematic 5. The Microchip WiFly module is an example of only one of the many modules we can plug into the XBee-compatible socket.

microSD socket is tied to the ATmega328P's SPI portal; microSD card insert on status is monitored by a MOSFET (FDN339N) that is triggered by the mechanical switch mounted within the microSD card socket.

The OpenLog firmware is available from links provided on the SparkFun site. You will need an AVR programmer to program the OpenLog firmware into the ATmega328P. I use an old (and apparently no longer sold) AVRISP mkII. You can find an assortment of AVR hardware programmers on the web, including a couple at SparkFun.

The ATmega328P programmer socket that is mounted on the original design is a six-pin 1.25 mm-pitch male manufacturer part number DF13-6P-1.25DSA.

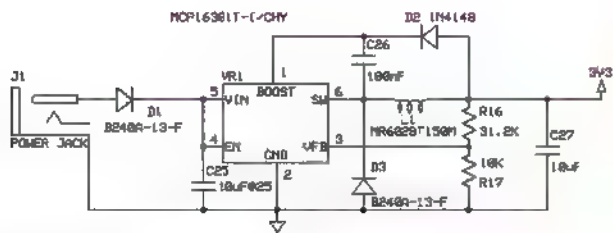
Since the original design's ExpressPCB files are available to you, you can modify the original design to meet your needs in this area.

### Design Details – RS-232

Very little rocket science exists in Schematic 4. Our RS-232 interface circuitry consists of the standard charge pump capacitors and a workhorse ST3232 RS-232 interface IC. The ST3232's RTS signal is hardwired in a DCE (Data Communications Equipment, i.e., modem device) configuration. Moving this signal to the ST3232's pin 10 forces a DTE (Data Terminal Equipment, i.e., terminal device) configuration.

### Design Details – Radio

The WiFly module is depicted in Schematic 5. In reality, this could be an XBee module, an ACKme Hopper, or any other XBee replacement radio. All of our target radio modules are identical in terms of their simple three-wire serial interface.



■ Schematic 6. This switcher is good for 600 mA. It generates very little heat and takes up very little space.

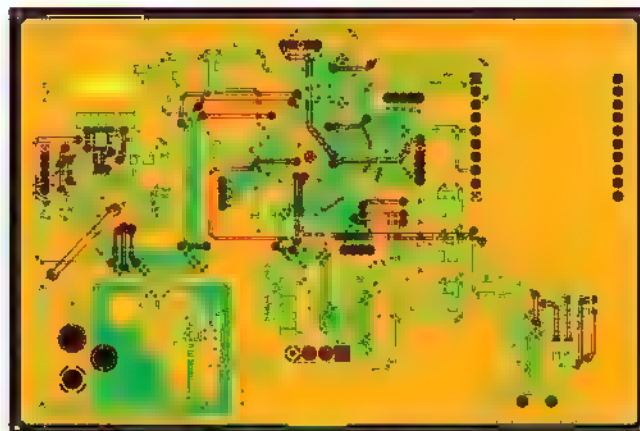
## Design Details – Power Supply

We've implemented a Microchip MCP16301T-based power supply circuit instead of a pair of capacitors and a three-terminal LDO voltage regulator IC stuck to a big copper pad. The MCP16301T high voltage input integrated switch step-down regulator is the prime player in our step-down voltage regulator configuration. All of the solid-state switching components are contained within the MCP16301T's tiny SOT-23-6 package. The desired +3.3 VDC output voltage is determined by the 31.2K/10K resistor pair.

## Printed Circuit Board

**Screenshot 1** is a screen capture of the initial PCB design. The design's subsystems are laid out in their own allotted copper domain. I purposefully didn't lay out any mounting holes. I'll leave those up to you.

Note that I've applied a generous amount of ground plane on both sides of the PCB. The ground plane serves



■ Screenshot 1. This is our PCB design. The components used were chosen to allow hand soldering techniques to be applied in its construction. All of the parts are readily available from vendors that advertise in *Nuts & Volts*.

two purposes. It helps reduce the system's electrical noise and it acts as a unifying ground conductor for the top and bottom board layers.

## Next Up

In the next installment of Design Cycle, we will mount up all of our components. Once the PCB is totally stuffed, we'll write some code to bring each of our multipurpose communications node's subsystems to life. **NV**

**NEW!**  
from **velleman**



**3D PRINTER KIT**  
Model K8200



Build-it-yourself 3D printer. Prints objects of maximum size of 20 x 20 x 20 cm using either PLA or ABS filament (3mm.) Extremely fast, reliable and precise even when printing at high speeds. Model K8200 is compatible with RepRap software and firmware (Free download). Sturdy aluminum construction, with heated print bed. Nine different colors of PLA filament available at \$39.00. Complete with power supply and all necessary parts. Filament sold separately.

**NOW ONLY \$795.00**  
Part No. 32VKK8200M

**Electronix EXPRESS**

TERMS Min \$20 + shipping School Purchase Orders, VISA MC Money Order Prepaid NO PERSONAL CHECKS. NO COD. NJ Residents Add 7% Sales Tax

In NJ: 732-381-8020  
FAX: 732-381-1006

980 Hart Street • Rahway, NJ 07065  
800-972-2225

<http://www.velleman.com>  
email: [electron@elexp.com](mailto:electron@elexp.com)



For complete product details, visit our webstore!

# The Nuts & Volts **WEBSTORE**

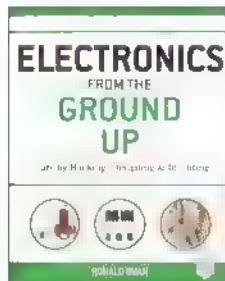
## GREAT FOR DIYers!

### Electronics from the Ground Up: Learn by Hacking, Designing, and Inventing

by  
Ronald Quan

Are you fascinated by the power of even the smallest electronic device? Electronics from the Ground Up guides you through step-by-step experiments that reveal how electronic circuits function so you can advance your skills and design custom circuits. You'll work with a range of circuits and signals related to optical emitters and receivers, audio, oscillators, and video.

Paper back 544 pages.  
**\$30.00**

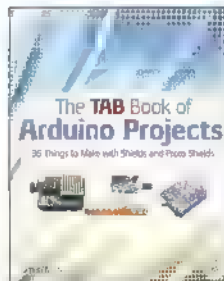


### The TAB Book of Arduino Projects by Simon Monk

The ultimate  
collection of DIY  
Arduino projects!

In this easy-to-follow book, electronics guru Simon Monk shows you how to create a wide variety of fun and functional gadgets with the Arduino Uno and Leonardo boards. Filled with step-by-step instructions and detailed illustrations, The TAB Book of Arduino Projects: 36 Things to Make with Shields and Proto Shields provides a cost estimate, difficulty level, and list of required components for each project.

**\$30.00**



### Arduino Projects for Amateur Radio

by Jack Purdum, Dennis Kidder

Boost Your  
Ham Radio's  
Capabilities Using  
Low Cost Arduino  
Microcontroller  
Boards

Do you want to increase the functionality and value of your ham radio without spending a lot of money? This book will show you how! *Arduino Projects for Amateur Radio* is filled with step-by-step microcontroller projects you can accomplish on your own — no programming experience necessary.

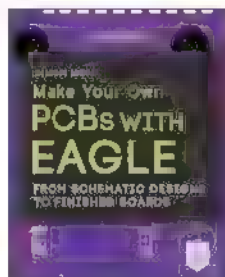
**\$30.00**



### Make Your Own PCBs with EAGLE by Eric Kleinert

Featuring detailed illustrations and step-by-step instructions, *Make Your Own PCBs with EAGLE* leads you through the process of designing a schematic and transforming it into a PCB layout. You'll then move on to fabrication via the generation of standard Gerber files for submission to a PCB manufacturing service. This practical guide offers an accessible, logical way to learn EAGLE and start producing PCBs as quickly as possible.

**\$30.00**



### Build Your Own Transistor Radios by Ronald Quan

A Hobbyist's Guide to High  
Performance and Low-Powered  
Radio Circuits

Create sophisticated transistor radios that are inexpensive yet highly efficient. Inside this book, it offers complete projects with detailed schematics and insights on how the radios were designed. Learn how to choose components, construct the different types of radios, and troubleshoot your work.

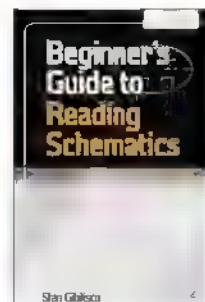
\*Paperback, 496 pages  
**\$49.95**



### Beginner's Guide to Reading Schematics, 3E by Stan Gibilisco

Navigate the roadmaps of simple electronic circuits and complex systems with help from an experienced engineer. With all-new art and demo circuits you can build, this hands-on, illustrated guide explains how to understand and create high-precision electronics diagrams. Find out how to identify parts and connections, decipher element ratings, and apply diagram-based information in your own projects.

**\$25.00**



### How to Diagnose and Fix Everything Electronic by Michael Jay Geier

Master the Art of  
Electronics Repair

In this hands-on guide, a lifelong electronics repair guru shares his tested techniques and invaluable insights. *How to Diagnose and Fix Everything Electronic* shows you how to repair and extend the life of all kinds of solid-state devices, from modern digital gadgetry to cherished analog products of yesteryear.

**\$24.95**



### Programming PICs in Basic by Chuck Hellebuyck

If you wanted to learn how to program microcontrollers, then you've found the right book! Microchip PIC microcontrollers are being designed into electronics throughout the world and none is more popular than the eight-pin version. Now the home hobbyist can create projects with these little microcontrollers using a low cost development tool called the CHIPAXE system and the Basic software language. Chuck Hellebuyck introduces how to use this development setup to build useful projects with an eight-pin PIC12F683 microcontroller. **\$14.95**



### Programming Arduino Next Steps: Going Further with Sketches by Simon Monk

In this practical guide, electronics guru Simon Monk takes you under the hood of Arduino and reveals professional programming secrets. Also shows you how to use interrupts, manage memory, program for the Internet, maximize serial communications, perform digital signal processing, and much more. All of the 75+ example sketches featured in the book are available for download. **\$20.00**



Order online @ [www.store.nutsvolts.com](http://www.store.nutsvolts.com)  
 Or CALL 1-800-783-4624 today!

**EDUCATIONAL**

**Beginners Guide Book Combo.**

**Only \$85.95**  
 Plus  
 FREE Priority Mail Shipping  
 US Only

**DO YOU LOVE RADIOS?**

**Sale Price \$39.95**  
**Sale Price \$19.95**  
**Sale Price \$23.95**

**Get 20% off these three books plus Free Std. Shipping!**

To order call 800 783-4624 or visit: <http://store.nutsvolts.com>

arduinoclassroom.com

*Arduino Classroom - learn computing and electronics*

The free Internet virtual textbook: Arduino 101 at [www.arduinoclassroom.com](http://www.arduinoclassroom.com) provides a sensible learning sequence that introduces computing and electronics with clear text and detailed hands-on labs with tested examples using the Arduino Projects Kit.

Available from Nuts&Volts for only \$44.99

**CD-ROM SPECIAL**

**Nuts & Volts  
 11 CD-ROMs  
 & Hat Special!**  
 That's 132 issues.  
 Complete with supporting  
 code and media files.

**Only \$229.95**  
 or \$24.95 each.

**The Nuts & Volts  
 Pocket Ref**

All the info you need at your fingertips!

This great little book is a concise all-purpose reference featuring hundreds of tables, maps, formulas, constants & conversions.

**Only \$12.95**

Visit <http://store.nutsvolts.com> or call (800) 783-4624



**CALL 1-800-783-4624 today!**  
**Order online @ [www.nutsvolts.com](http://www.nutsvolts.com)**

**PROJECTS**

**Wireless Freezer Alarm**



Looking for a way to protect all your frozen yummys??? In the June 2015 issue, we have an article that will help you do just that.

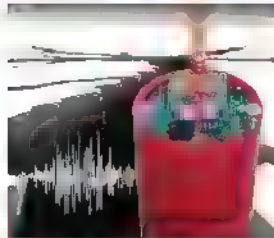
The kit includes:

- | Transmitter PCB, | Receiver PCB
- | Transmitter Programmed Chip
- | Receiver Programmed Chip

Other components can be found at your favorite parts house.

**\$39.95**

**Seismograph Kit**



Now you can record your own shaking, rattling, and rolling.

The Poor Man's Seismograph is a great project/device to record any movement in an area where you normally shouldn't have any. The kit includes everything needed to build the seismograph. All you need is your PC, SD card, and to download the free software to view the seismic event graph.

**\$79.95**

**3D LED Cube Kit**



This kit shows you how to build a really cool 3D cube with a 4 x 4 x 4 monochromatic LED matrix which has a total of 64 LEDs. The preprogrammed microcontroller that includes 29 patterns that will automatically play with a runtime of approximately 6-1/2 minutes. Colors available: Green, Red, Yellow & Blue

**\$67.95**

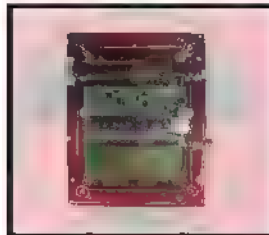
**Solar Charge Controller Kit 2.0**



If you charge batteries using solar panels, then you can't afford not to have them protected from over-charging. This 12 volt/12 amp charge controller is great protection for the money. It is simple to build, ideal for the novice, and no special tools are needed other than a soldering iron and a 9/64" drill!

**\$27.95**

**Geiger Counter Kit**



This kit is a great project for high school and university students. The unit detects and displays levels of radiation, and can detect and display dosage levels as low as one micro-roentgen/hr. The LND 712 tube in our kit is capable of measuring alpha, beta, and gamma particles.

*Partial kits also available.*

**\$159.95**

**Super Detector Circuit Set**



Pick a circuit!

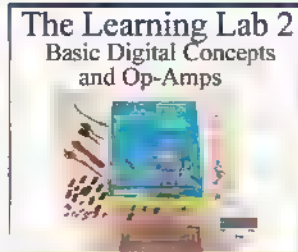
With one PCB you have the option of detecting wirelessly: temperature, vibration, light, sound, motion, normally open switch, normally closed switch, any varying resistor input, voltage input, mA input, and tilt, just to name a few.

**\$32.95**

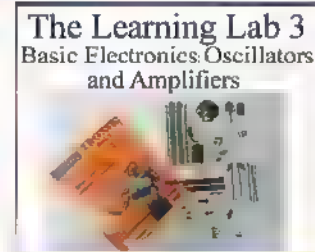
**FOR BEGINNER GEEKS!**



**\$59.95**



**\$49.95**



**\$39.95**

These labs from LF Components show simple and interesting experiments and lessons, all done on a solderless circuit board.

As you do each experiment, you learn how basic components work in a circuit, and continue to build your arsenal of knowledge with each successive experiment.

**For more info and lab details, please visit our webstore.**

# CLASSIFIEDS

## NEW PRODUCTS

Continued from page 21

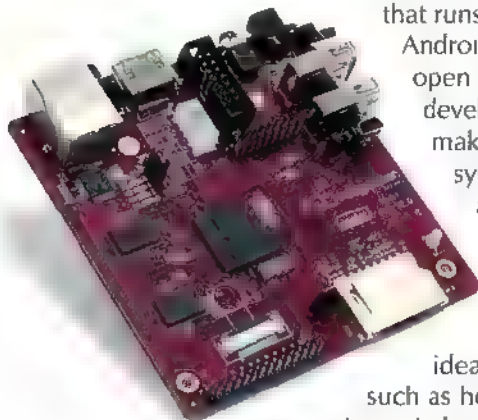
telecom (e.g., AC/DC and DC/DC power supplies), industrial (e.g., solar inverters, LED lighting, HID lighting, battery chargers, projectors, and welders), and automotive (e.g., LED and HID headlights, DC/DC converters), among others.

The dsPIC33EP family is supported by Microchip's MPLAB® Starter Kit for Digital Power (part #DM330017-2, \$129.99) which allows customers to explore using the new GS family in popular digital power-conversion topologies.

For more information, contact:  
**Microchip**  
[www.microchip.com](http://www.microchip.com)

## MIPS-BASED MICRO COMPUTER

Imagination Technologies announces an updated version of their MIPS-based microcomputer called Creator CI20. Launched in December 2014, the Creator CI20 is an



affordable microcomputer that runs Linux and Android, and enables open source developers, the maker community, system integrators, and others to implement a wide array of applications quickly. It is ideal for projects such as home automation, gaming, wireless multimedia

streaming, and others that require a high performance Linux or Android platform with GPU and video capabilities, and provides excellent connectivity.

This new version includes an improved board layout that optimizes Wi-Fi performance, and is easier to mount in cases. They've also designed a new 3D printable enclosure. Builders can use the source files available on Imagination Technologies' website to build their own versions. On the software side, they are adding out-of-the-box FlowCloud support, and also bringing new features to Linux and Android:

- FlowCloud is an IoT (Internet of Things) API designed by Imagination to connect devices to the cloud; it already runs on other MIPS-based dev boards, including

Continued on page 81

## SURPLUS

### SURPLUS ELECTRONIC PARTS & ACCESSORIES



Over 20,000 Items In Stock

Belts  
Cables  
Connectors  
Fans

Hardware  
LEDs  
Motors  
Potentiometers

Relays  
Semiconductors  
Service Manuals  
Speakers

Switches  
Test Equipment  
Tools  
VCR Parts

Surplus Material Components  
**SMC ELECTRONICS**  
[www.smcelectronics.com](http://www.smcelectronics.com)

No Minimum Order  
Credit Cards and PAYPAL Accepted.  
Flat \$4.95 per order USA Shipping.

## KITS/PLANS

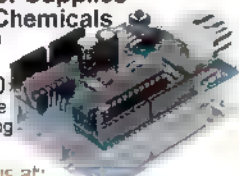


**QKITS LTD**  
sales@qkits.com

**1 888 GO 4 KITS**

Arduino • Raspberry Pi  
Power Supplies  
MG Chemicals  
RFID

\$8.50  
flat rate  
shipp.ng

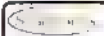


Visit us at:  
[www.qkits.com](http://www.qkits.com)

## LIGHTING



## SERVICES



**Sandpiper Electronics Services LLC**

design of custom electronics instrumentation  
Circuit design & drawings PCB layout fabrication & population  
Prototype development for research, industry & inventors  
39 years experience in electronics development for research & flight applications laser systems & laboratory research

FREE no obligation consultations  
[www.sandtronics.com](http://www.sandtronics.com)

## ROBOTICS

### Need sensors?



[www.maxbotix.com](http://www.maxbotix.com)

build robots?  
Then, you need  
a subscription  
to **SERVO!**

[www.servomagazine.com](http://www.servomagazine.com)

Subscribe today!  
[www.servomagazine.com](http://www.servomagazine.com)

## WIRE/CABLE



**ANAHEIM WIRE, INC.**  
Master distributor of  
electrical and electronic  
wire and cable since  
**1973**. Items available from  
stock: Hook up wire, Shrink  
tubing, Cable ties, Connectors,  
etc. Wire cut & strip to specs,  
twisting, stripping. If interested,  
please call **1-800-626-7540**,  
FAX: 714-563-8309.  
Visa/MC/Amex **See us on  
the Internet:** [www.anaheimwire.com](http://www.anaheimwire.com)  
or email: [info@anaheimwire.com](mailto:info@anaheimwire.com)

## MISC FOR SALE

**SPEAKER BUSINESS CLOSED  
AFTER 35 YEARS.**  
SELLING \$250,000 INVENTORY OF  
SPEAKERS, SPEAKER PARTS AND  
MANUFACTURING EQUIPMENT AT A  
FRACTION OF COST. ALL OFFERS  
CONSIDERED. ISE, SAN BENITO, TX.  
Info [www.lselliquidator.com](http://www.lselliquidator.com)  
956-444-0004, 888-351-5550

## HARDWARE WANTED

**DEC EQUIPMENT  
WANTED!!!**  
Digital Equipment Corp.  
and compatibles.  
Buy - Sell - Trade  
CALL KEYWAYS 937-847-2300  
or email [buyer@keyways.com](mailto:buyer@keyways.com)



## >>> QUESTIONS

### PIR Hookup

My home alarm system has a motion sensor that has failed and the alarm company wants \$89 for a new one! I removed the bad one and it has screw terminals labeled:

- \* GND
- \* 12V
- \* ALARM COM
- \* ALARM NC
- \* TAMP1
- \* TAMP2

The simple PIRs I find for use with the Arduino are 5V and they don't have "tamp" pins. Can someone provide a schematic on how to hook up one of these low cost replacements?

**#7151** Christopher Randazzo  
Worcester, MA

### Tie Breaker Circuit

Does anyone have a circuit for a homemade "tie breaker" system? A teacher at my son's school is having a quiz contest where kids have to "buzz in." I need to build a circuit that can indicate who pressed a button first. I would prefer to use simple electrical components for this project as I am not really adept at programming microcontrollers.

**#7152** Alex Ferguson  
Pittsburgh, PA

### TV Headphone Amp

I listen to TV at night using headphones. I have noticed that the volume level that can be produced

by the TV set is actually too low for me to clearly make out the dialog. Is there a booster amp that has a tone control that I could use between the TV and my headphones to make the sound louder and clearer? I would like to build this myself so a schematic would be appreciated.

**#7153** Zachary Mailey  
Palmdale, CA

## >>> ANSWERS

### Voltage Reduction

*What would be the most efficient way to reduce the voltage from a nine volt battery to 5V? I could use a 7805 but it seems to bleed off a lot of power as heat. Is there a more efficient circuit or part?*

**#1** You might try using an LDO regulator like the LM2596S chip. I bought a 1.5V-37V DC/DC buck converter from MPJA for \$1.95 to reduce 12V to 9V, and it runs very cool.

Gene Sellier  
Fairhope, AL

**#2** Yes, the 7800 and 7900 series seem to just dissipate heat to drop the voltage. Try the LM2940-5. I found them at Jameco for \$1.39 each.

Schneids  
via email

**#3** The UA78S40 is an old switching regulator that is obsolete, but parts are available on eBay. The datasheet has a schematic for 26

volts to 10 volts, but you only need to change two resistors for five volts output: R1 = 30K, R2 = 10K.

A more modern solution is the LM2576 buck regulator. It only uses two capacitors, a diode, and an inductor for external components. The datasheet has a schematic that you can use directly.

The package is TO-220-5, (Mouser part number is: 926-LM2576T-5.0/nopb, cost: \$2.82). The IC is good for three amps but if your load is less than 20 mA, it would not be a good choice.

Russell L. Kincaid  
Milford, NH

**#4** A very economical solution are the LM25XX switching regulators on a board, which sell cheap from MPJA or the Internet. I also use the MC34063 which is often found in the "12 volt to five volt" cell phone adapters at your local thrift store, but easier to work with purchased new.

For low current applications, my favorites are the encapsulated 1W or 3W switching DC-DC regulators from CUI, Murata, etc., via Digi-Key

Jim Lacenski  
Bellevue, WA

**#5** Take a look into DC-DC buck converters. Many of the C manufacturers offer low part count devices, and depending on your load requirements, may offer a completely integrated solution.

Things to consider when selecting your buck converter:

1. Internal switch/es to the IC – ease of implementation.
2. Synchronous design – higher efficiency.
3. Load capabilities – This is one of the most important things to consider, especially with an integrated design. You want to select an IC that meets your load requirements, i.e., output current, but not something that blows them out of the water.

In other words, If your output

All questions AND answers are submitted by *Nuts & Volts* readers and are intended to promote the exchange of ideas and provide assistance for solving technical problems. All submissions are subject to editing and will be published on a space available basis if deemed suitable by the publisher. Answers are submitted

by readers and **NO GUARANTEES WHATSOEVER** are made by the publisher. The implementation of any answer printed in this column may require varying degrees of technical experience and should only be attempted by qualified individuals.

*Always use common sense and good judgment!*

Send all questions and answers by email to [forum@nutsvolts.com](mailto:forum@nutsvolts.com) or via the online form at [www.nutsvolts.com/tech-forum](http://www.nutsvolts.com/tech-forum)

requirements are 5V at 100 mA (0.5W), don't choose a device capable of 10W, or 2A at 5V. The reason being is these devices are more efficient when operated at the loads they were designed for.

One last thing. Review the EVKIT the manufacturer has available for the device. These are really good starting points for layout and design with the device. I hope this helps.

Justin J  
King City, CA

**[#5153 - May 2015]**

**On/Off Circuit**

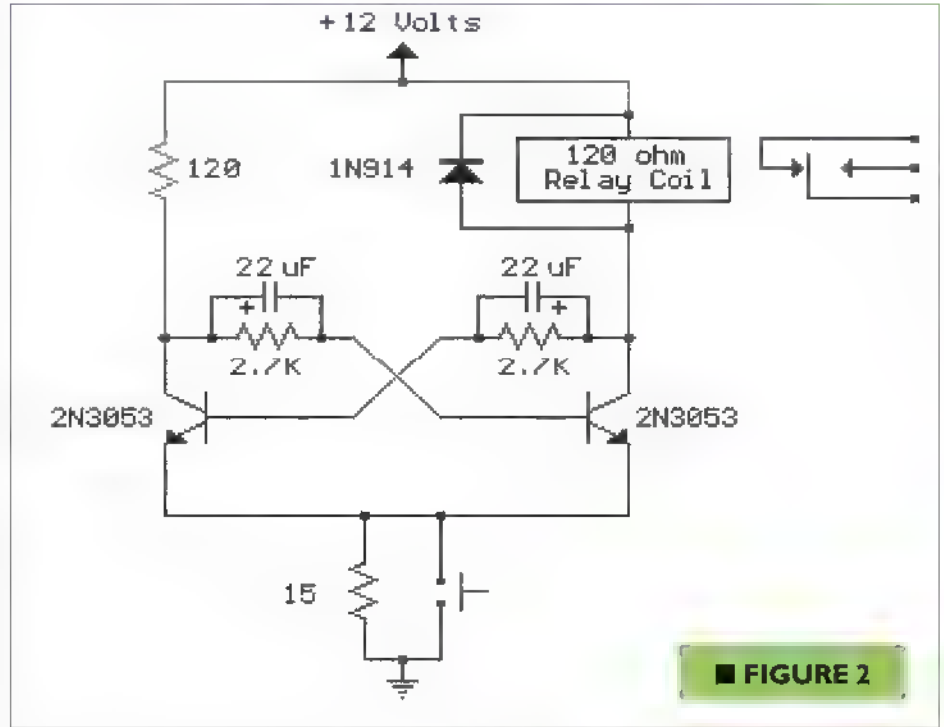
*Is there a simple circuit that would allow a normally open pushbutton to turn on a relay when pressed, and then turn it off when pressed again?*

**#1** Here are two circuits that should do what you want: Grove-2 @10 from [www.seeedstudio.com](http://www.seeedstudio.com) (YES, there are three e's); Cebek 1-9 @ \$4 from [www.mcmelectronics.com](http://www.mcmelectronics.com).

Also, there are a variety of electro-mechanical relays available at many sources. These do not need a circuit to latch but when the relay is activated, it toggles a switch from off to on.

Michael Herman  
La Quinta, CA

**#2** It's funny you should post this request. I just happen to have an article submitted to *Nuts & Volts* utilizing this exact same situation



**FIGURE 2**

that will hopefully be appearing in an upcoming issue!

In the meantime, **Figure 1** is the (modified) relevant portion of the schematic as you requested. The 74LS109 is a JK positive-edge triggered flip-flop. The input is normally tied to ground, but when you press the switch, that input is momentarily brought high, triggering the flip-flop, and toggling the relay between its on and off states. R2 and C1 are to debounce the switch and prevent multiple false inputs.

Derek Tombrello  
Columbiana, AL

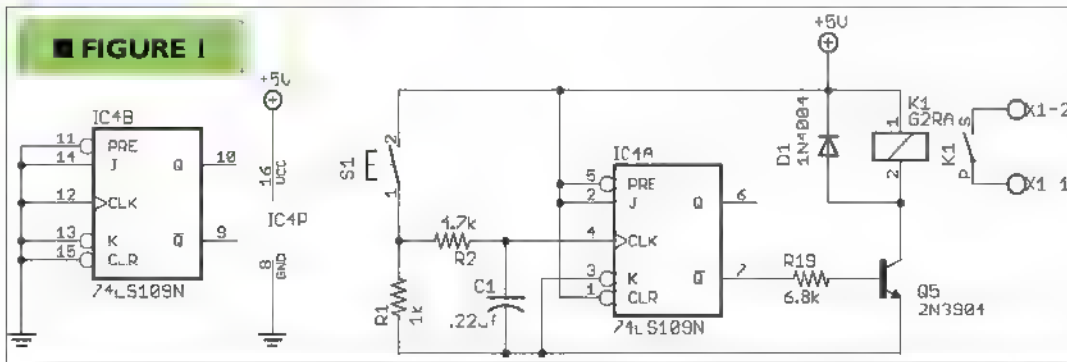
**#3** Look into a switch debouncer such as the MAX6816 and JK, or T flip-flops to drive the gate of a low side switch on your relay. The output of the switch debouncer could drive the clock of a T, or a properly configured JK flip-flop. Don't forget the protective diode across the relay in order to clamp the voltage when turning the relay on/off. I hope this helps.

Justin J  
King City, CA

**#4** Figure 2 is a simple circuit that can be used to toggle a relay using one switch. A relay with a 12 VDC is used, however, the relay contacts can control a larger voltage if needed.

Craig Kielhofer  
Wheeling, IL

**#5** It can't get any simpler than two resistors, one capacitor, and the relay.



**FIGURE 1**

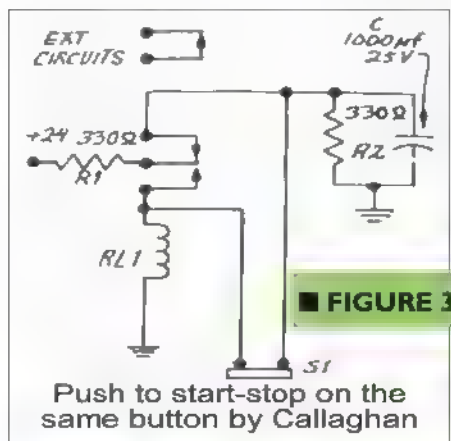


When the circuit in **Figure 3** is first energized, C charges to  $1/2 V_{cc}$  through voltage divider R1/R2. When the button S1 is pushed, C discharges through the relay coil. It pulls in, and the relay is held closed by the current through R1 and the relay coil. When the button is pushed again, C, (now fully discharged by R2), goes across the relay coil making it drop out and returning the circuit to where it started

**Mike Callaghan**  
La Crescenta, CA

**#6** Use a pushbutton switch to toggle a JK flip-flop.

**Lance Corey**  
via email



**[#5154 - May 2015]**  
**Fuse Confusion**

*I have a drill press that has a five amp fuse that recently has started to blow on just about every project. I am not working on harder material and the bits are sharp, so I don't think its increased torque from friction which leaves the electronics. As this is simply a motor and a power switch, I can't imagine what might be causing the increased current draw. Suggestions?*

**#1** You don't say what type of fuse you are using, but if it is a buss type glass fuse you might want to make sure you are using a slow-blow type fuse for your drill press.

**Gene Sellier**  
Fairhope, AL

**#2** Worn bearings could be allowing the armature to rub on the stator. Remove the belt and try to move the motor shaft; if it moves at all, it is bad

**Russell L. Kincaid**  
Milford, NH

**#3** If this is just an on/off switched drill press, then you definitely have an excessive friction issue. If a good lubrication doesn't help, then the

brushes or commutator in the motor are worn and binding.

**Michael Herman**  
La Quinta, CA

**#4** Does the dill press sound any different? Maybe the bearing in the motor is wearing out, causing the increased load.

What about the cord? Has it been pinched/damaged in any way? Or even the outlet that it is plugged into, are there visible signs of scorching?

Lastly, how old is it? Could the motor be electrically failing? Things to consider beyond stresses due to operator use.

**Justin J**  
King City, CA

**#5** Mr. Olivo is experiencing nuisance fuse blowing in his drill press. The problem may well be that the fuses that he has been using cannot tolerate the inrush current demand of the motor. At the instant of power application, the magnetic state of the motor core may be such that the first half-cycle of utility power drives it into saturation.

Assuming that the fuse is a  $1/4"$  x  $1-1/4"$  cartridge fuse, a Bussmann MDL-5 fuse will open at five amperes,

but will tolerate more than 100 amperes during a one-half cycle of utility power. See the time-current curves for this family on datasheet #2004, which is downloadable from [www.cooperindustries.com/content/public/en/bussmann/electrical/products/electronic\\_smalldimension/elx\\_1\\_4\\_x\\_1-1\\_4\\_/mdl-v\\_mdl.catalog\\_numbers\\_\(amps\).brands.cooper\\_bussmann.html](http://www.cooperindustries.com/content/public/en/bussmann/electrical/products/electronic_smalldimension/elx_1_4_x_1-1_4_/mdl-v_mdl.catalog_numbers_(amps).brands.cooper_bussmann.html).

If the fuse is another size, check with your local electrical jobber for an appropriate fuse having similar inrush tolerance. Good luck.

**Peter A. Goodwin**  
Rockport, MA

## JOIN TEAM SYNERGY MOON!

Synergy Space Explorers are the power that drives the Synergy Moon space program. We are Team Synergy Moon, building a mission to the moon that includes Micro Satellites, Lunar Rovers and a Lunar Lander. We're going into space this year, and we're going to the moon with our Google Lunar XPRIZE mission.

You will have the opportunity to participate in space research, exploration and development missions, which currently include our Google Lunar XPRIZE mission to the moon, the Artemis NanoSat Constellation project and our remote controlled Tesla Orbital Space Telescope.

Get on board now and be part of this great adventure!

DIY SATELLITES AND SPACECRAFT SYSTEMS  
[info@synergymoon.com](mailto:info@synergymoon.com)



## NEW PRODUCTS

Continued from page 77

the chipKIT Wi-FIRE from Digilent. FlowCloud gives users comprehensive device-to-cloud infrastructure and services, enabling IoT innovators to rapidly create new applications based on the CI20 such as home automation, robotics, industrial monitoring and control, and many others.

- For Android 4.4, several improvements have been made, including audio over HDMI and Bluetooth; new built-in Ethernet settings; audio jack auto-detection (easily switch audio output from HDMI to headphones, and vice versa); and audio recording. Support for USB storage is coming soon.

- For Linux, Imagination is working on updating to kernel version 3.18 which will offer a boost in vital performance areas such as memory speed and graphics.

The new MIPS Creator CI20 incorporates an Ingenic JZ4780 SoC which includes a 1.2 GHz dual-core MIPS32 processor and PowerVR SGX540 GPU. Retail price is \$65.

For more information, contact:  
**Imagination Technologies**  
[www.imgtec.com](http://www.imgtec.com)

LOOK FOR SEARCH FOR FIND

Find your favorite advertisers here!

# Advertiser INDEX

### AMATEUR RADIO

#### ANDTV

National RF .....21  
 Palstar .....64  
 Ramsey Electronics .....82-83

### BATTERIES/CHARGERS

Cunard Associates .....21  
 HiTec. ....2

### BUYING ELECTRONIC SURPLUS

All Electronics Corp. ....6  
 Earth Computer Technologies 65  
 Weirdstuff Warehouse .....21

### CCD CAMERAS/VIDEO

Ramsey Electronics .....82-83

### CIRCUIT BOARDS

AP Circuits .....33  
 Cunard Associates .....21  
 Dimension Engineering .....7  
 ExpressPCB .....10  
 Front Panel Express LLC .....11  
 Saelig Co. Inc. ....11

### COMPONENTS

All Electronics Corp. ....6  
 Electronix Express .....73

### COMPUTER

#### Hardware

Earth Computer Technologies 65  
 Weirdstuff Warehouse .....21

### Microcontrollers /

#### I/O Boards

Images Co. ....21  
 M.E. Labs .....39  
 MikroElektronika .....3  
 Technologic Systems .....25

### DESIGN/ENGINEERING/ REPAIR SERVICES

Cleveland Institute of Electronics.33  
 ExpressPCB .....10  
 Front Panel Express LLC .....11  
 Images Co. ....21  
 National RF .....21

### EDUCATION

Boxed Kit Amps .....21  
 Cleveland Institute of Electronics.33  
 Command Productions .....65  
 NKC Electronics .....21  
 Poscope.....17

### ENCLOSURES

Front Panel Express LLC .....11

### EVENTS

BattleBots .....Back Cover

### HI-FI AUDIO

Boxed Kit Amps .....21

### KITS & PLANS

Boxed Kit Amps .....21  
 Earth Computer Technologies 65

NKC Electronics .....21

Ramsey Electronics .....82-83

### MISC./SURPLUS

All Electronics Corp. ....6  
 Front Panel Express LLC .....11  
 Weirdstuff Warehouse .....21

### PROGRAMMERS

M.E. Labs .....39  
 MikroElektronika .....3

### RF TRANSMITTERS/ RECEIVERS

National RF .....21

### ROBOTICS

BattleBots .....Back Cover  
 Cleveland Institute of Electronics.33  
 HiTec. ....2  
 Robot Power .....21

### TEST EQUIPMENT

Dimension Engineering .....7  
 Images Co. ....21  
 NKC Electronics .....21  
 Poscope.....17  
 Saelig Co. Inc. ....11

### TOOLS

MikroElektronika .....3  
 PanaVise .....5  
 Poscope.....17

All Electronics Corp. ....6

AP Circuits .....33

BattleBots .....Back Cover

Boxed Kit Amps .....21

Cleveland Institute of

Electronics .....33

Command Productions .....65

Cunard Associates .....21

Dimension Engineering.....7

Earth Computer Technologies 65

Electronix Express .....73

ExpressPCB .....10

Front Panel Express LLC ....11

HiTec. ....2

Images Co. ....21

M.E. Labs.....39

MikroElektronika .....3

National RF.....21

NKC Electronics .....21

Palstar .....64

PanaVise .....5

Poscope.....17

Ramsey Electronics .....82-83

Robot Power .....21

Saelig Co. Inc. ....11

Technologic Systems .....25

Weirdstuff Warehouse .....21



# Beat The Heat...Build a Kit!

## Ramsey Kits Are Always Cool, Even In The Summer Heat!

### Electrocardiogram ECG Heart Monitor

- ✓ Visible and audible display of your heart rhythm!
- ✓ Bright LED "Beat" indicator for easy viewing!
- ✓ Re-usable hospital grade sensors included!
- ✓ Monitor output for professional scope display
- ✓ Simple and safe 9V battery operation

When we think summer, we normally think of vacations, traveling, and all the activities you have been waiting for all winter! And whether that includes hiking that new trail you've heard about or simply riding the new rides (and waiting in line in the scorching heat!) at Wally World, there WILL be physical exertion involved! While we are frequently reminded that February is national Heart Smart month, we think every month should be Heart Smart month. Heart Smart is a way of life, and certainly shouldn't be limited to one month a year. We kept that in mind when we designed the ECG!

Not only will building an actual ECG be a thrill, but you'll get hands-on knowledge of the relationship between electrical activity and the human body. Each time the human heart beats, the heart muscle causes small electrical changes across your skin. By monitoring and amplifying these changes, the ECGIC detects the heartbeat and allows you to accurately display it, and hear it, giving you a window into the inner workings of the human heart and body!

Use the ECGIC to astound your physician with your knowledge of ECG/EKG systems. Enjoy learning about the inner workings of the heart while, at the same time, covering the stage-by-stage electronic circuit theory used in the kit to monitor it. The three probe wire pick-ups allow for easy application and experimentation without the cumbersome harness normally associated with ECG monitors.

The documentation with the ECGIC covers everything from the circuit description of the kit to the circuit description of the heart! Multiple "beat" indicators include a bright front panel LED that flashes with the actions of the heart along with an adjustable level audio speaker output that supports both mono and stereo hook-ups. In addition, a monitor output is provided to connect to any standard oscilloscope to view the traditional style ECG/EKG waveforms just like you see in a real ER or on one of the medical TV shows!



Look what I found!

On a personal note... See the display to the left? That's me! In between writing this monthly ad copy, catalog copy, and plethora of other tasks here, I noticed some skipped beats in my pulse! An immediate cardiac check found I had something called Trigeminy, or PVCs that occur at intervals of 2 normal beats to one PVC! And I saw it with our ECG1 kit!

The fully adjustable gain control on the front panel allows the user to custom tune the differential signal picked up by the probes giving you a perfect reading and display every time! 10 hospital grade re-usable probe patches are included together with the matching custom case set shown. Additional patches are available in 10-packs.

Operates on a standard 9VDC battery (not included) for safe and simple operation. Note, while the ECGIC professionally monitors and displays your heart rhythms and functions, it is intended for hobbyist usage only. If you experience any cardiac symptoms, seek proper medical help immediately!

ECG1C	Electrocardiogram Heart Monitor Kit With Case & Patches	\$44.95
ECG1WT	Electrocardiogram Heart Monitor, Factory Assembled & Tested	\$89.95
ECG10	Electrocardiogram Re-Usable Probe Patches, 10-Pack	\$4.95

### Digital Voice Changer

This voice changer kit is a riot! Just like the expensive units you hear the DJ's use, it changes your voice with a multitude of effects! You can sound just like a robot, you can even ad vibrato to your voice! 1.5W speaker output plus a line level output! Runs on a standard 9V battery.

MK171 Voice Changer Kit \$9.95

### Precision PC Plane Antennas

Our LPY series PC antennas continue to be the favorite for virtually all RF and wireless applications. From microwave links, wireless mics, to RFID, we've got you covered. Check our site for details!

LPYSeries Precision PC Plane Antennas from \$29.95

### Laser Trip Senser Alarm

True laser protects over 500 yards! At last within the reach of the hobbyist, this neat kit uses a standard laser pointer (included) to provide both audible and visual alert of a broken path. 5A relay makes it simple to interface! Breakaway board to separate sections.

LTS1 Laser Trip Sensor Alarm Kit \$29.95

### Steam Engine & Whistle

Simulates the sound of a vintage steam engine locomotive and whistle! Also provides variable "engine speed" as well as volume, and at the touch of a button the steam whistle blows! Includes speaker. Runs on a standard 9V battery.

MK134 Steam Engine & Whistle Kit \$11.95

### Audio Recorder & Player

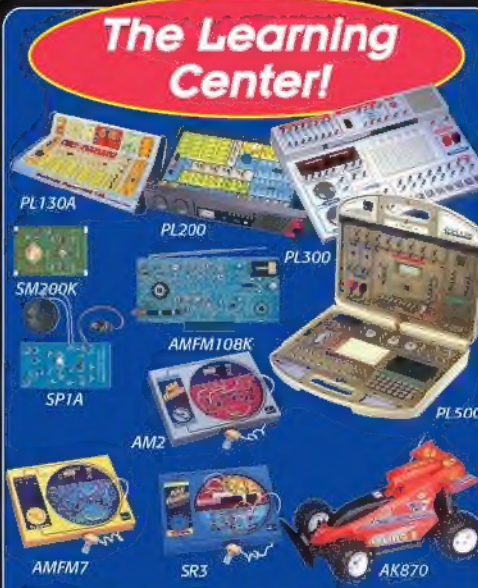
Record and playback up to 8 minutes of messages from this little board! Built-in condenser mic plus line input, line & speaker outputs. Adjustable sample rate for recording quality. 4-switch operation that can be remote controlled! Runs on 9-12VDC at 500mA.

K8094 Audio Recorder/Player Kit \$32.95

### Stereo Ear Super Audio Amp

This "Stereo Ear" amp is one of the neatest and handiest high gain amps you will find! Dual high sensitivity electret mics are amplified 50x to provide the ultimate stereo source! Output is 3.5mm jack, runs on three AAA's.

MK136 Stereo Ear Audio Amp Kit \$9.95



### Beginners To Advanced... It's Fun!

- ✓ Learn, build, and enjoy!
- ✓ 130, 200, 300, & 500 in one electronic labs!
- ✓ Practical through hole and SMT soldering labs!
- ✓ Integrated circuit AM/FM radio lab!
- ✓ AM, AM/FM, and SWL radio labs!
- ✓ Radio Controlled (RC) car!
- ✓ Beginner's non-soldering kits!

For over 4 decades we've become famous for making electronics fun, while at the same time making it a great learning experience. As technology has changed over these years, we have continued that goal!

**PL130A** Gives you 130 different electronic projects together with a comprehensive learning manual describing the theory behind all the projects.

**PL200** Includes 200 very creative fun projects and includes a neat interactive front panel with 2 controls, speaker, LED display and a meter.

**PL300** Jump up to 300 separate projects that start walking you through the learning phase of digital electronics.

**PL500** The ultimate electronics lab that includes 500 separate projects that cover it all, from the basics all the way to digital programming.

**SP1A** Whether young or old, there's always a need to hone your soldering skills. Either learn from scratch or consider it a refresher, and end up with a neat little project when you're done!

**SM200K** Move up to Surface Mount Technology (SMT) soldering, and learn exactly how to solder those tiny little components to a board!

**AMFM108K** We not only take you through AM and FM radio theory but we guide you through IC's. When you're done you've built yourself an IC based AM/FM radio that works great!

**AM2** Learn the complete theory of AM broadcast radio and end up with a highly sensitive AM radio receiver!

**AMFM7** Step up to AM/FM with this multi-band lab and learn the basics of both bands. The FM tuner is factory assembled and aligned!

**SR3** Enter the world of SWL with this short-wave radio learning lab covering 6-8MHz and 12-18MHz!

**AK870** One of the most exciting electronic learning kits that the kids will love! Build a complete RC speedster from the ground up! 7 remote functions!

PL130A	130-In-One Lab Kit	\$39.95
PL200	200-In-One Lab Kit	\$84.95
PL300	300-In-One Lab Kit	\$109.95
PL500	500-In-One Lab Kit	\$249.95
SP1A	Through Hole Soldering Lab	\$9.95
SM200K	SMT Practical Soldering Lab	\$22.95
AMFM108K	AM/FM IC Lab Kit & Course	\$36.95
AM2	AM Radio Learning Lab	\$11.95
AMFM7	AM/FM Radio Learning Lab	\$12.95
SR3	Short Wave Radio Learning Lab	\$16.95
AK870	RC Speedster Car Kit	\$29.95



There's only so much room on these two pages, so check it all out in our new virtual electronic catalog! Flip through the pages and search with ease! Visit [www.ramseycatalog.com](http://www.ramseycatalog.com)

**Follow Us and SAVE \$\$**   
Follow us on your favorite network site and look for a lot of super deals posted frequently... exclusively for our followers!



## Stereo Audio Platform Gain Controller

- ✓ Stereo audio processing while preserving audio dynamics!
- ✓ True stereo control keeps virtual sonic source location intact!
- ✓ Auto-bypass restores original levels when power is turned off!
- ✓ Built-in bar graph indication of signal level with display mute!

The SGC1 is one of our latest kits, and provides a great solution to the age-old problem: how can we easily correct inconsistent audio levels without negatively affecting the dynamics of the audio signal? The SGC1 circuit implements a principle known as the "Platform Gain Principle," which was originally developed by CBS Labs to allow transmitted audio levels to be automatically adjusted to keep them within a desired range.

Think of it like an audio engineer, constantly adjusting the output level in order to limit highs that would be too loud while boosting lower levels so that they can still be heard. You may think "oh, this is just another limiter/compressor!" Not so! Here's the real trick: keeping the full dynamic range ratio of the output signal the same as the original input - something the typical limiter/compressor can only dream of doing! The SGC1 can be placed in just about any standard analog stereo line level audio circuit to keep the audio level within the desired range. It's also the perfect addition to any of our hobby kit transmitters, allowing you to match levels between different audio sources while keeping lows audible and preventing the highs from overdriving.

In addition to its useful basic function and great audio performance, the SGC1 also boasts a front panel LED meter to give an indication of the relative level of the input signal, as well as a front level control that allows you to adjust the controller to the min/max center point of your desired level range. And yes, it is a **Stereo Gain Controller!** Meaning that the levels of both the left and right channels are monitored and adjusted equally, thereby maintaining the relative virtual position of things like instruments, singers and speakers! The entire unit is housed in a slim attractive black textured aluminum case that is sure to complement your studio or home theatre. If you're looking for perfect audio levels, hire a broadcast audio engineer, but if that doesn't fit your budget, the SGC1 is the next best thing! Includes 15VDC world-wide power adapter.



## Synthesized FM Stereo Transmitter Kit

The FM25 has been the hobbyist's standard for FM synthesized stereo transmitters for more than two decades! Just plug in the stereo left/right audio from your MP3 player, CD player, or computer, and broadcast it on any frequency in the standard FM broadcast band.

The FM25B brings stereo line level audio into the unit through a 3.5mm stereo phone jack. A loop out is provided at the input for monitoring. The unit features a PIC microprocessor for easy frequency programming through board mounted DIP switches. The transmit frequency is Phase Locked Loop (PLL) controlled for unparalleled stability making frequency drift a thing of the past, extremely critical for digital tuners. The RF output level is adjustable from 5uW to 25 mW via a potentiometer. Use the built-in whip antenna or use an external antenna with the standard "F" external antenna connector on the rear panel. Includes power supply & stereo cable. (Note: The FM25B is a do-it-yourself learning kit that you assemble. Please remember that the end user is responsible for complying with all FCC rules & regulations within the US, or any regulations of their respective governing body in regards to the application and use of the FM25B.)



SGC1 Stereo Audio Platform Gain Controller Kit \$179.95

## RF Preamp

The famous RF preamp that's been written up in the radio & electronics magazines! This super broadband preamp covers 100 KHz to 1000 MHz! Unconditionally stable gain is greater than 16dB while noise is less than 4dB! 50-75 ohm input. Runs on 12-15 VDC.



SA7 RF Preamp Kit \$16.95

## Mad Blaster Warble Alarm

If you need to simply get attention, the "Mad Blaster" is the answer, producing a LOUD ear shattering raucous racket! Super for car and home alarms as well. Drives any speaker. Runs on 9-12VDC.



MB1 Mad Blaster Warble Alarm Kit \$9.95

## Water Sensor Alarm

This little \$7 kit can really "bail you out"! Simply mount the alarm where you want to detect water level problems (sump pump)! When the water touches the contacts the alarm goes off! Sensor can even be remotely located. Runs on a standard 9V battery.



MK108 Water Sensor Alarm Kit \$6.95

## Air Blasting Ion Generator

Generates negative ions along with a hefty blast of fresh air, all without any noise! The steady state DC voltage generates 7.5kV DC negative at 400uA, and that's LOTS of ions! Includes 7 wind tubes for max air! Runs on 12-15VDC.



IG7 Ion Generator Kit \$64.95

## Tri-Field Meter Kit

"See" electrical, magnetic, and RF fields as a graphical LED display on the front panel! Use it to detect these fields in your house, find RF sources, you name it. Featured on CBS's Ghost Whisperer to detect the presence of ghosts! Req's 4 AAA batteries.



TFM3C Tri-Field Meter Kit \$74.95

## Electret Condenser Mic

This extremely sensitive 3/8" mic has a built-in FET preamplifier! It's a great replacement mic, or a perfect answer to add a mic to your project. Powered by 3-15VDC, and we even include coupling cap and a current limiting resistor! Extremely popular!



MC1 Mini Electret Condenser Mic Kit \$3.95

## Touch Switch

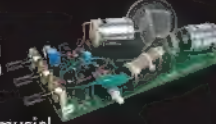
Touch on, touch off, or momentary touch hold, it's your choice with this little kit! Uses CMOS technology. Actually includes TWO totally separate touch circuits on the board! Drives any low voltage load up to 100mA. Runs on 6-12 VDC.



TS1 Touch Switch Kit \$9.95

## Laser Light Show

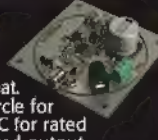
Just like the big concerts, you can impress your friends with your own laser light show! Audio input modulates the laser display to your favorite music! Adjustable pattern & speed. Runs on 6-12VDC.



LLS1 Laser Light Show Kit \$49.95

## 20 Watt Mini Audio Amp

Delivers a super clean 20W output from one SMT package! Ultra efficient class D design produces no heat. PCB can be snapped into a small circle for special applications. Runs on 18VDC for rated output, or down to 10VDC for reduced output.



UAM2 20W Subminiature Amp Kit \$52.95

## Tickle-Stick Shocker

The kit has a pulsing 80 volt tickle output and a mischievous blinking LED. And who can resist a blinking light and an unlabeled switch! Great fun for your desk, "Hey, I told you not to touch!" Runs on 3-6 VDC.



TS4 Tickle Stick Kit \$9.95

## Passive Aircraft Monitor

The hit of the decade! Our patented receiver hears the entire aircraft band without any tuning! Passive design has no LO, therefore can be used on board aircraft! Perfect for airshows, hears the active traffic as it happens! Available kit or factory assembled.



ABM1 Passive Aircraft Receiver Kit \$89.95

## 12VDC Regulated Supply

Go green with our new 12VDC 1A regulated supply. Worldwide input 100-240VAC with a Level-V efficiency! It gets even better, includes DUAL ferrite cores for RF and EMI suppression. All this at a 10 buck old wallwart price!



AC121 12VDC 1A Regulated Supply \$9.95

FM25B Synth FM Stereo Xmtr Kit \$139.95

## 3.4dB Gain FM Omni Antenna

Our 5/8 wave omni antenna has been the standard for LPFM installations worldwide. Provides 3.4dB gain while keeping the signal radiation low to the horizon for maximum range. Field tuneable over the entire FM range for a perfect match. SO239 connector.



FMA200E Omnidirectional FM Antenna \$139.95

## LED Red Blinky

Alternately flashes two jumbo red LEDs. You've seen this used on name badges, hats, buttons, and attention getters for decades! Has been the most popular 1st learning kit for kids young and old! Runs on 3-9VDC.



BL1 LED Red Blinky Kit \$7.95

## HV Plasma Generator

Generate 2" sparks to a handheld screwdriver! Light fluorescent tubes without wires! This plasma generator creates up to 25kV at 20kHz from a solid state circuit! Build plasma bulbs from regular bulbs and more! Runs on 16VAC or 5-24VDC.



PG13 HV Plasma Generator Kit \$64.95

## Speedy Speed Radar Gun

Our famous Speedy radar gun teaches you doppler effect the fun way! Digital readout displays in MPH, KPH, or FPS. You supply two coffee cans! Runs on 12VDC or our AC121 supply.



SG7 Speed Radar Gun Kit \$74.95

## Broadband RF Preamp

Need to "perk-up" your counter or other equipment to read weak signals? This preamp has low noise and yet provides 25dB gain from 1MHz to well over 1GHz. Output can reach 100mW! Runs on 12 volts AC or DC or the included 110VAC PS. Asmb.



PR2 Broadband RF Preamp \$69.95

## 12VDC Worldwide Supply

It gets even better than our AC121 to the left! Now, take the regulated Level-V green supply, bump the current up to 1.25A, and include multiple blades for global country compatibility! Dual ferrite cores!



PS29 12VDC 1.25A Worldwide Supply \$19.95

GET THE NUTS & BOLTS DISCOUNT!

Mention or enter the coupon code **NVRMZ142** and receive **10% off your order!**


**800-446-2295**  
**www.ramseykits.com**

**RAMSEY ELECTRONICS®**

590 Fishers Station Drive  
Victor, NY 14564  
(800) 446-2295  
(585) 924-4560

Prices, availability, and specifications are subject to change. We are not responsible for typos, stupid, printer's bleed, or 4th of July fireworks accidents! In the true holiday spirit, Robin made me do this ad copy instead of going on the family picnic! Not Fair! Visit www.ramseykits.com for the latest pricing, specials, terms and conditions. Copyright 2015 Ramsey Electronics™...so there!



A promotional poster for the TV series BattleBots. The background is a bright, golden-yellow sky with a large, dark, mechanical claw-like structure framing the top and sides. In the center, three people (a woman, a man, and a boy) are standing in a boxing ring, cheering with their arms raised. The man in the center has his arms raised high. The woman on the left is also cheering. The boy on the right is holding a small robot. The overall scene is set in a large arena with a crowd of spectators visible in the background.

Who will build  
the ultimate fighting machine?

# BATTLEBOTS

**NEW SERIES**  
**JUNE 21 SUNDAYS 9|8c**



#BattleBots